UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
ENGENHARIA DE COMPUTAÇÃO

ANDRÉ AUGUSTO DEQUECH CARVALHO
FABIANO GUILHERME PRADO ARAÚJO
PAULO BERTOLDI RENAUX
RAFAEL G. TURCHENSKI

ROBÔ MAPEADOR

RELATÓRIO TÉCNICO

CURITIBA
2013

ANDRÉ AUGUSTO DEQUECH CARVALHO
FABIANO GUILHERME PRADO ARAÚJO
PAULO BERTOLDI RENAUX
RAFAEL G. TURCHENSKI

ROBÔ MAPEADOR

Monografia apresentada à disciplina Oficina de Integração 3, no curso de Engenharia de Computação da Universidade Tecnológica Federal do Paraná.

Professores: João A. Fabro
Heitor S. Lopes

CURITIBA
2013

Resumo

CARVALHO, André A. D.; ARAÚJO, Fabiano G. P.; RENAUX, Paulo B.; TURCHENSKI, Rafael G. 2013. Robô Mapeador. Relatório Técnico – Programa de Graduação em Engenharia de Computação, Universidade Tecnológica Federal do Paraná. Curitiba, 2013.

Este trabalho descreve as etapas efetuadas para o planejamento e execução do projeto "Robô Mapeador". O objetivo deste projeto é o desenvolvimento de um sistema computacional que mapeia de forma estimada o percurso efetuado por um robô criado pela equipe. Este robô tem o movimento controlado de duas formas distintas: controlado remotamente pelo usuário na estação-base ou de modo automático. No primeiro modo, o robô obedece comandos de movimentação enviados por uma estação base. Enquanto no segundo modo, o robô percorre um circuito constituído de uma faixa preta em um fundo branco de forma autônoma, utilizando como base informações de uma câmera de linha. O mapeamento é baseado em informações geradas por um giroscópio e um sensor de efeito hall, que atua como um encoder. Como resultado obteu-se um sistema capaz de realizar as funções descritas.

Palavras-chave: Freescale, Robô Mapeador, Odometria, Sistema Embarcado, Estação Base.

Abstract

This paper describes the tasks completed in the planning and execution of the project entitled "Mapper Robot". The goal of this project is developing a computational system capable of mapping the trajectory of a robot developed by the team. The robot is controlled by two distinct ways: manually controlled by the user acessing a base station or automatically controlled. In the first way, the robot follows commands sent by the base-station while in the second one, the robot follows a circuit composed of a black line in a white background autonomously, using as a reference the data gathered by a line scan camera. The mapping is based on a gyroscope and a hall effect sensor, that acts as an encoder. The result of this project is a computational system that performs the functions described.

Key-Words: Freescale, Mapper Robot, Odometry, Embedded Systems, Base Station.

Lista de Figuras

Lista de Tabelas

Sumário

1 — Introdução

1.1 Motivação

O projeto teve como inspiração a competição internacional realizada anualmente pela empresa de semicondutores Freescale. Entitulada "Copa Freescale", esta competição tem como objetivo a construção de um carro autônomo capaz de percorrer um circuito constituído de uma faixa preta em um fundo branco no menor tempo possível. O hardware a ser utilizado será o mesmo presente no kit da competição para realização de uma função distinta, o mapeamento do percurso completado pelo carro. Acreditamos que este objetivo é extremamente rico pois expõe a equipe a vários problemas de engenharia como o interfaceamento de sensores, circuitos de potência, protocolos de comunicação, controle de posição, entre outros.

Além das diversas áreas de conhecimento abrangidas pelos requisitos do projeto, o mapeamento relizado pode gerar dados para otimizar algoritmos de controle de robôs e a documentação gerada no projeto para a utilização do hardware será de grande valia para outros alunos que queiram participar da competição.

1.2 Trabalhos correlatos

O projeto é baseado na plataforma utilizada para a competição copa Freescale. Esta competição ocorre mundialmente todos os anos e em 2012, só na Europa, Oriente Médio e África contava com mais de 70 times e 200 estudantes. Nesta seção há uma breve análise das duas principais abordagens utilizadas para a competição.

Como sensor principal para navegação tipicamente utiliza-se a câmera de linha ou sensores infravermelho (IR). A primeira alternativa fornece uma precisão de controle maior, no entanto obtêm-se um ciclo de aquisição e processamento de dados maior e uma sensibilidade à iluminação externa considerável. A abordagem utilizando um módulo de sensores fotoelétricos é mais robusta no quesito interferência de iluminação externa, no entanto fornece uma leitura da pista mais próxima, dificultando a capacidade de prever o comportamento da pista e tomar ações anteriormente à variação no percurso.[2].

Além da construção de todo o módulo de controle autônomo também será criado um modo em que o robô é controlado diretamente pelo o usuário, o que é muito semelhante ao realizado pelo projeto Bellator, implementado na disciplina em semestres anteriores.

1.3 Objetivos

1.3.1 Objetivos do Robô

- O robô deverá ser capaz de percorrer de forma autônoma uma pista construída segundo os critérios estabelecidos pela copa Freescale.
- O robô deverá conter um modo de operação controlado por usuário, onde utilizará os comandos fornecidos pela base para movimentação.
- O robô deverá processar os dados vindos da câmera e dos sensores infravermelho e controlar sua direção e velocidade para efetuar o trajeto da pista.

- O robô deverá conter os seguintes sensores instalados: um acelerômetro, um giroscópio, encoders nas rodas, uma câmera de linha.
- O robô deverá ser capaz de transmitir as informações adquiridas pelos sensores para uma estação base localizada a menos de 30m de sua posição, utilizando um módulo de transmissão Wireless.
- O robô deverá ser capaz de receber e interpretar comandos que alterem sua orientação e sua velocidade.

1.3.2 Objetivos do Software da Estação Base

- O software deverá plotar para o usuário a trajetória estimada percorrida pelo robô, utilizando para isto informações dos sensores que o carro possui.
- O software deverá fornecer ao usuário a opção de enviar comandos que alterem a direção e a velocidade do carrinho, possibilitando o controle remoto do robô pelo usuário.
- O software deverá indicar para o usuário a velocidade do carro e a estimativa de sua orientação
- O software deverá mostrar para o usuário o status da conexão com o robô e alertar em caso de falha na conexão.
- O software deverá alterar o modo de funcionamento do carro, possibilitando a troca entre modo autônomo e modo controlado por usuário.

2 — Planejamento

2.1 Premissas e Restrições

2.1.1 Premissas

Serão especificadas, nesta seção, as premissas adotadas pela equipe para o desenvolvimento do projeto.

- Todos os componentes necessários (kit de desenvolvimento Freescale, sensores e outros componentes eletrônicos e mecânicos) estarão nas mãos da equipe até o dia 30 de julho de 2013.
- Durante o período no qual a equipe ainda não tiver obtido e montado o kit, ou então caso este seja extraviado durante o desenvolvimento do projeto, teremos acesso a outro robô Freescale emprestado por uma equipe que já participou da competição em anos anteriores.
- A equipe possui os conhecimentos e habilidades necessários para a realização do projeto, ou então possui a capacidade de adquirir tais habilidades e conhecimentos a tempo de realizar o projeto.

2.1.2 Restrições

Serão especificadas, nesta seção, os motivos que podem limitar as possibilidades de escolha da equipe para o projeto:

- O tempo disponível para a etapa de execução do projeto é de apenas 10 semanas, do dia 31/07/2013 ao dia 09/10/2013. A data de prazo final é inegociável.
- O custo máximo de componentes para o projeto é de 900 reais.
- O ambiente pelo qual o robô irá se deslocar será uma pista nos moldes da competição copa Freescale.
- O comprimento das diagonais da pista não será maior que vinte metros.
- O ambiente onde a pista estará será iluminado, com uma luminosidade equivalente a ao período diurno de um dia.
- O robô utilizará apenas os dados da câmera. As informações dos outros sensores serão utilizadas apenas pela estação base, para estimar o posicionamento do carro.

2.2 Custo do Projeto

Nesta seção será apresentado os custos detalhados do projeto. Tendo concluído a realização da análise tecnológica, a equipe já definiu os componentes necessários para a implementação do projeto e a quantidade de horas estimada.

O custo do projeto foi dividido em duas categorias distintas: gastos com componentes (hardware/software) e gastos com mão-de-obra, estimando para o último caso o salário de R$35,00 por hora trabalhada para cada integrante da equipe. As projeções de custo para as duas categorias podem ser vistas na tabela 1 e tabela 2, respectivamente. A contagem de horas trabalhadas foi iniciada após a entrega do Project Charter, onde é considerado o início

do projeto. Na tabela 1 não foi levado em conta a possibilidade do abatimento dos custos pela obtenção de material emprestado.

No desenvolvimento do projeto foram utlizados apenas softwares freeware ou que já foram adquiridos anteriormente por membros da equipe para a redução de custos.

Tabela 1: Tabela de custos dos componentes do projeto.

Quantidade	Componente	Custo Estimado
1	Kit de desenvolvimento Freescale (Placa de desenvolvimento MPC5604B) (Chassi do carro) (Servomotor Futaba S-3010) (Ponte H dupla com máx. 5A) (Câmera de linha TAOS 1401)	R$292,00
2	Encoder	R$40,50
1	Acelerômetro e giroscópio (MPU-6050)	R$50,00
1	Bateria VEX 7,2V 2000mA	R$89,00
2	Módulo RF nRF24L01	R$ 20,00
1	Servomotor Turginy TG9	R$12,00
1	Placa de suporte de acrílico	R$30,00
1	Fabricação da PCB de interfaceamento	R$100,00
1	Placa de LEDs	R$60,00
Custo Total	R$693,50	

Tabela 2: Tabela dos custos com mão-de-obra do projeto.

Quantidade de Horas Trabalhadas	Custo
520	R$18.200,00

O desenvolvimento do projeto deverá ocorrer dentro do prazo de 10 semanas, com início no dia 10/07, quando ocorre a finalização da etapa de planejamento e fim no dia 25/09, data final da entrega do projeto concluído e da monografia. A figura 3 apresenta as principais etapas do projeto.

Tabela 3: Principais etapas do projeto.

Principais Fases	Data	Custos
Compra de todos os componentes.	26/06 a 30/07	R$855,00
Planejamento do projeto.	26/06 a 10/07	R$4.200,00
Desenvolvimento do primeiro *deliverable*.	10/07 a 07/08	R$4.200,00
Desenvolvimento do segundo *deliverable*.	07/08 a 21/08	R$2.800,00
Entrega do terceiro *deliverable*.	21/08 a 04/09	R$2.800,00
Entrega do quarto *deliverable*.	04/09 a 18/09	R$2.800,00
Entrega do projeto concluído e da documentação.	18/09 a 25/09	R$1.400,00

2.3 Formação da Equipe e Escolha do Gerente

A equipe foi formada respeitando o sorteio feito em sala de aula na disciplina de Oficinas de Integração 3, pelos professores João A. Fabro e Heitor S. Lopes.
Gerente: Fabiano Guilherme Prado Araújo.
Colaboradores: André Augusto Dequech Carvalho, Paulo Bertoldi Renaux, Rafael Góes Turchenski.

2.4 Responsabilidades e Autoridades do Gerente

2.4.1 Responsabilidades do Gerente

Dentre as responsabilidades do gerente, destacam-se:

- O gerente deverá controlar o cronograma do projeto, estabelecendo prazos e metas para o cumprimento de tarefas no projeto.
- O gerente será o responsável pela alocação das tarefas existentes para os membros da equipe.
- O gerente deverá assessorar o desenvolvimento das tarefas alocadas para os integrantes da equipe, visando garantir uma sinergia e o bem estar dos integrantes no grupo.
- O gerente deverá ser o agente comunicador entre os membros do grupo e garantir que exista uma comunicação clara e eficaz entre os integrantes da equipe e entre a equipe e os professores.
- O gerente também executará funções técnicas no projeto, não se limitando apenas à atividade de gerência, recebendo, entretanto, um menor volume de trabalho do que o alocado aos outros integrantes.
- O gerente deverá ser o representante da equipe no projeto e, portanto, deverá responder pela equipe sempre que for solicitado.
- O gerente ficará responsável por fiscalizar o cumprimento das tarefas alocadas para os integrantes do projeto, assim como por assegurar a qualidade dos trabalhos realizados.

2.4.2 Autoridade do Gerente

- O gerente será quem terá a palavra final sobre quaisquer decisões efetuadas pela equipe, ficando responsável por quaisquer decorrências destas decisões. Deverá, entretanto, respeitar e levar em conta a opinião da equipe.
- O gerente poderá alocar as tarefas que devem ser executadas no projeto de forma que lhe melhor aprouver entre os integrantes da equipe.
- O gerente poderá avaliar o desempenho dos integrantes da equipe com consequências diretas na nota da disciplina.

2.5 Planejamento de Riscos

1ª etapa: **Identificação do Risco**

Denominação do risco: Subestimar o Tempo Necessário para o Cumprimento de uma Tarefa	**Nº Identificação** 1

Descrição do risco: A complexidade e o tempo do cumprimento de uma tarefa ou etapa de projeto pode ser subestimado pela equipe.

2ª etapa: **Avaliação do Risco**

Impacto: ○ 5 (alto) **● 4 (médio/alto)** ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) Pode ocasionar uma demanda de trabalho excessiva para a equipe, aumentando o nível de stress do grupo e em casos graves gerando atrasos no cronograma.
Probabilidade: **● 5 (alto)** ○ 4 (médio/alto) ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) Como o grupo não possui experiência prévia no desenvolvimento de sistemas embarcados e nem com módulos de comunicação Wireless, não é possível saber de forma precisa a complexidade das etapas que terão de ser desenvolvidas no decorrer do projeto. Visto que o número de tarefas que nunca foram desenvolvido anteriormente é grande, subestimar ao menos uma tarefa é praticamente certo.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Questionar outras pessoas que já executaram uma tarefa semelhante em outros projetos, de forma a realizar um melhor estimativa. Pesquisar o máximo possível sobre os passos e o procedimento de execução da tarefa, de forma a obter uma visão clara de como e do que deve ser feito. Mitigar: Acrescentar ao tempo estimado de uma tarefa uma margem de erro, de forma a amenizar os efeitos do erro de julgamento.	
Impacto reavaliado: 3 (médio)	**Probabilidade reavaliada: 3 (médio)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Falha no gerenciamento de tempo em semana de provas	**Nº Identificação** 2
Descrição do risco: Diminuição no ritmo de trabalho da equipe não esperada causada por uma semana de provas.	

2ª etapa: **Avaliação do Risco**

Impacto: ○ 5 (alto) ○ 4 (médio/alto) ● **3 (médio)** ○ 2 (médio/baixo) ○ 1 (baixo) A diminuição não planejada no volume de trabalho da equipe em uma semana de provas pode gerar um atraso no andamento do projeto.
Probabilidade: ○ 5 (alto) ● **4 (médio/alto)** ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) A ocorrência deste risco depende da atenção do grupo na hora de efetuar a construção do cronograma.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Anotar em uma agenda o calendário de provas de todos os integrantes da equipe, alocando uma quantidade menor de trabalho para períodos em que haja avaliações.	
Impacto reavaliado: 2 (médio/baixo)	**Probabilidade reavaliada: 3 (médio)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Falta de comunicação entre integrantes da equipe	**Nº Identificação** 3
Descrição do risco: A comunicação ineficiente entre os membros da equipe.	

2ª etapa: **Avaliação do Risco**

Impacto: ○ 5 (alto) ○ 4 (médio/alto) **● 3 (médio)** ○ 2 (médio/baixo) ○ 1 (baixo)
Uma comunicação ineficiente entre os membros da equipe pode gerar confusão no estabelecimento de prazos e nas divisões de tarefas, prejudicando a sincronização da equipe e o progresso do projeto como um todo, além de gerar discórdia. Outro impacto da falta de comunicação é a dificuldade em acompanhar o que cada membro está fazendo, o que dificulta a obtenção de uma visão geral do progresso do projeto e da fase em que ele se encontra.
Probabilidade: ○ 5 (alto) ○ 4 (médio/alto) **● 3 (médio)** ○ 2 (médio/baixo) ○ 1 (baixo)
A ausência de comunicação é o padrão, caso não haja medidas que impeçam que isso ocorra. Consideramos então razoável a chance de termos um problema de comunicação na equipe.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Dividir o grupo em subgrupos de mais de um integrante, para que haja uma cobrança entre o próprio subgrupo, descentralizando o papel do gerente como controlador exclusivo do desenho das tarefas e gerando uma maior comunicação entre o grupo. Mitigar: Realizar reuniões semanais que obriguem a cada integrante a fornecer um panorama geral das atividades executadas na semana anterior, e das dificuldades encontradas. Obrigar os integrantes a manterem um histórico diário das atividades desenvolvidas.	
Impacto reavaliado: 2 (médio/baixo)	**Probabilidade reavaliada: 2 (médio/baixo)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Não cumprimento do prazo para a entrega da tarefa	**Nº Identificação** 4
Descrição do risco: O prazo determinado pelo gerente para entrega de determinada tarefa é excedido.	

2ª etapa: **Avaliação do Risco**

Impacto: ○ 5 (alto) • **4 (médio/alto)** ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) Pode atrasar o desenvolvimento de alguma outra tarefa e até inviabilizar o projeto.
Probabilidade: • **5 (alto)** ○ 4 (médio/alto) ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) A impossibilidade do gerente saber de antemão as habilidades e o conhecimento prévio de cada membro da equipe sobre áreas do saber necessários para a execução do projeto pode ocasionar a geração de um cronograma que não é realizável para um determinado membro. Outros riscos podem, também podem causar este. Como este risco já ocorreu até o desenvolvimento deste documento, estimamos que ocorra em etapas posteriores do projeto com uma alta probabilidade.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Acompanhamento da execução das tarefas dos membros da equipe, implementar uma comunicação eficaz na equipe, auxiliar membros com dificuldades técnicas. Mitigar: Inserir uma margem maior de tempo para integrantes que não tenham conhecimento prévio do assunto da tarefa.	
Impacto reavaliado: 3 (médio)	**Probabilidade reavaliada: 5 (alto)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Redução da equipe por meio de desistência de algum integrante	**Nº Identificação** 5
Descrição do risco: Desistência de seguir adiante com a disciplina por algum membro da equipe.	

2ª etapa: **Avaliação do Risco**

Impacto: • **5 (alto)** ○ 4 (médio/alto) ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) A falta de um integrante pode ser crucial para equipe pois a redução da mão-de-obra disponível na equipe pode sobrecarregar o trabalho dos outros integrantes, atrasando ou até inviabilizando o projeto.
Probabilidade: ○ 5 (alto) ○ 4 (médio/alto) ○ 3 (médio) • **2 (médio/baixo)** ○ 1 (baixo) Contando com o histórico da disciplina em outros semestres, verificamos que a taxa de desistência dos alunos é baixa.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Manter um ambiente de trabalho estimulante e de fácil convivência. Buscar motivar sempre que possível os integrantes. Mitigar: Buscar gerar redundância de conhecimento entre os membros da equipe, minimizando perdas em caso de desistência. Aceitar: Refazer o cronograma do projeto, contando com o integrante a menos	
Impacto reavaliado: 4 (médio/alto)	**Probabilidade reavaliada: 1 (baixo)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Redução momentânea da equipe por problemas de saúde	**Nº Identificação** 6
Descrição do risco: Afastamento temporário de um integrante por problemas de saúde.	

2ª etapa: **Avaliação do Risco**

Impacto: ○ 5 (alto) ○ 4 (médio/alto) ○ 3 (médio) ● **2 (médio/baixo)** ○ 1 (baixo)
A redução momentânea do grupo pela saída de um integrante pode atrasar alguma parte do projeto, dificultando o cumprimento do cronograma. Como doenças raramente são capazes de abater um indivíduo por dois ou mais dias, o impacto deste risco é considerado baixo.
Probabilidade: ○ 5 (alto) ○ 4 (médio/alto) ● **3 (médio)** ○ 2 (médio/baixo) ○ 1 (baixo)
A chance de que algum membro da equipe fique doente no decorrer do semestre é alta, mas a chance de que a doença adquirida seja forte o suficiente para prejudicar o desempenho do membro no projeto é menor.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Tentar desenvolver um ambiente tão livre quanto possível de stress e um cronograma organizado para evitar que os membros sofram restrições de sono e/ou alimentação e tenham o sistema imunológico prejudicado	
Impacto reavaliado: 2 (médio/baixo)	**Probabilidade reavaliada: 2 (médio/baixo)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Gasto maior que o estimado com o projeto	**Nº Identificação** 7
Descrição do risco: O custo do projeto ultrapassou a quantia estimada de uma forma significativa.	

2ª etapa: **Avaliação do Risco**

Impacto: • **5 (alto)** ∘ 4 (médio/alto) ∘ 3 (médio) ∘ 2 (médio/baixo) ∘ 1 (baixo) Um custo de projeto maior que o limite estipulado pelos sponsors pode inviabilizar o projeto.
Probabilidade: ∘ 5 (alto) ∘ 4 (médio/alto) • **3 (médio)** ∘ 2 (médio/baixo) ∘ 1 (baixo) Uma subestimação do custo pode ocorrer se os componentes necessários escolhidos pela equipe não estiverem disponíveis, e as opções de substituição forem muito caras.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Realizar a escolha de componentes que estejam disponíveis em pelo menos 2 fornecedores e que de preferência possam ser comprados no Brasil. Contabilizar no orçamento custos associados aos componentes como o frete e impostos, em caso de importação. Realizar uma pesquisa de mercado. Mitigar: Tentar conseguir componentes emprestado como backup, reduzindo o preço do projeto em caso de quebra de algum componente.	
Impacto reavaliado: 4 (médio/alto)	**Probabilidade reavaliada: 2 (médio/baixo)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Atraso no recebimento de componentes	**Nº Identificação** 8
Descrição do risco: Atraso no recebimento de componentes necessários para a execução do projeto.	

2ª etapa: **Avaliação do Risco**

Impacto: ● **5 (alto)** ○ 4 (médio/alto) ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo)
Pode atrasar a execução de tarefas, postergar a familiarização dos integrantes com os componentes necessários para a realização do projeto e atrasar o cronograma.
Probabilidade: ○ 5 (alto) ○ 4 (médio/alto) ● **3 (médio)** ○ 2 (médio/baixo) ○ 1 (baixo)
O atraso na entrega de componentes pode ser causado por um atraso na alfândega, problema em correios ou falta de disponibilidade em estoque, o que não são acontecimentos muito raros.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Obter por empréstimo os equipamentos necessários, para que o desenvolvimento do projeto não seja totalmente impedido pela falta de componentes. Mitigar: Encomendar os componentes necessários para o projeto o mais rápido possível, minimizando os efeitos dos atrasos.	
Impacto reavaliado: 4 (médio/alto)	**Probabilidade reavaliada: 3 (médio)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Problemas mecânicos com a inserção do encoder no carrinho	**Nº Identificação** 9
Descrição do risco: Problema com a instalação de um encoder no carro.	

2ª etapa: **Avaliação do Risco**

Impacto: ○ 5 (alto) • **4 (médio/alto)** ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) A estrutura das rodas do carro pode dificultar a instalação de um encoder, ou o chassi pode impedir o acesso às rodas, gerando transtornos para a equipe e ocupando muito tempo.
Probabilidade: ○ 5 (alto) ○ 4 (médio/alto) • **3 (médio)** ○ 2 (médio/baixo) ○ 1 (baixo) A equipe não conhece a estrutura interna do carrinho.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Procurar módulos de encoder que sejam compatíveis com a roda fornecida pelo Kit Freescale. Aceitar: A instalação de um encoder é importante para o projeto e as complicações oriundas da instalação terão que ser resolvidas pela equipe.	
Impacto reavaliado: 4 (médio/alto)	**Probabilidade reavaliada: 2 (médio/baixo)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Recebimento de componentes sem condições de uso	**Nº Identificação** 10
Descrição do risco: Algum componente do carrinho pode ter sofrido dano no transporte ou ter vindo queimado.	

2ª etapa: **Avaliação do Risco**

Impacto: ○ 5 (alto) ● **4 (médio/alto)** ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) Dependendo do componente danificado, o carrinho pode não funcionar gerando a necessidade de um novo pedido, o que pode atrasar substancialmente o projeto.
Probabilidade: ○ 5 (alto) ○ 4 (médio/alto) ○ 3 (médio) ● **2 (médio/baixo)** ○ 1 (baixo) As chances de ocorrerem danos em cargas transportadas pelo correio e do envio de componentes eletrônicos estragados é baixa.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Mitigar: Encomendar o carrinho o mais cedo possível permitindo tempo para efetuar um novo pedido se necessário. Obter peças emprestadas para que o projeto não seja interrompido caso o risco ocorra.	
Impacto reavaliado: 3 (médio)	**Probabilidade reavaliada: 1 (baixo)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Problemas referentes a implementação do protocolo de rede sem fio	**Nº Identificação** 11
Descrição do risco: Podem ocorrer problemas técnicos na utilização dos módulos de comunicação sem fio, usados para prover a interação entre o robô e a estação base.	

2ª etapa: **Avaliação do Risco**

Impacto: ○ 5 (alto) **● 4 (médio/alto)** ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) Problemas técnicos na comunicação podem atrasar o andamento das tarefas e, caso não sejam solucionados até a data de entrega inviabilizam o projeto.
Probabilidade: ○ 5 (alto) **● 4 (médio/alto)** ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) A equipe não possui conhecimento prévio sobre módulos de comunicação, nem sobre formas de implementação da comunicação.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Escolher um módulo de comunicação que seja fácil de trabalhar e que seja bem documentado.	
Impacto reavaliado: 3 (médio)	**Probabilidade reavaliada: 3 (médio)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Danos ou extravio da placa de desenvolvimento	**Nº Identificação** 12
Descrição do risco: Ocorrência de danos à placa de desenvolvimento que a inutilizem, ou o extravio da mesma.	

2ª etapa: **Avaliação do Risco**

Impacto: • **5 (alto)** ○ 4 (médio/alto) ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) A placa de desenvolvimento é o núcleo controlador do robô e sem dúvida o componente mais crítico do projeto. Sem ele, nenhum desenvolvimento no robô é possível, tanto na parte de sensores quanto na parte de atuadores e comunicação. Sem esta placa certamente teríamos um atraso no cronograma.
Probabilidade: ○ 5 (alto) ○ 4 (médio/alto) ○ 3 (médio) • **2 (médio/baixo)** ○ 1 (baixo) A possibilidade da ocorrência de danos à placa de desenvolvimento é real e maximizada por precisarmos transportar o carrinho rotineiramente.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Emprestar ou comprar outra placa de desenvolvimento como backup. Evitar o transporte do carrinho quando possível. Mitigar: Pedir a outro membro da equipe que inspecione o circuito construído antes da alimentação da placa.	
Impacto reavaliado: 4 (médio/alto)	**Probabilidade reavaliada: 2 (médio/baixo)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

1ª etapa: **Identificação do Risco**

Denominação do risco: Problema com a integração de dados dos sensores	**Nº Identificação** 13
Descrição do risco: A equipe pode ter dificuldades para encontrar alguma forma de integrar os dados vindos dos diversos sensores instalados no carro, de modo a oferecer uma interpretação coerente da movimentação do robô.	

2ª etapa: **Avaliação do Risco**

Impacto: ○ 5 (alto) ● **4 (médio/alto)** ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) Encontrar uma forma de interpretar os sensores de forma coerente é crucial para obtermos a estimativa da trajetória do carro, o que afetará diretamente a precisão do mapeamento realizado. A tarefa de integrar dados de diversos sensores para obtermos uma representação útil é, contudo, um problema aberto e constitui uma área de pesquisa. Dificuldades de encontrar um algoritmo razoável e implementável para estimarmos a trajetória podem atrasar o projeto.
Probabilidade: ● 5 (alto) ○ 4 (médio/alto) ○ 3 (médio) ○ 2 (médio/baixo) ○ 1 (baixo) Como nós estamos trabalhando com dados vindos de diversos sensores o problema de interpretá-los e utilizarmos as informações obtidas para tentarmos estimar a trajetória é certo.

3ª etapa: **Desenvolvimento da Resposta ao Risco**

Ações, Responsáveis e Datas de Conclusão	
Estratégias e ações: Prevenir: Iniciar a pesquisa de técnicas para realizar a integração dos sensores o mais cedo possível. Iniciar o desenvolvimento do sistema de mapeamento na estação-base o mais cedo possível, utilizando dados simulados. Mitigar: Realizar diversas voltas com o carro na pista durante o mapeamento, de forma a tentar eliminar os erros gradativamente. Utilizar conhecimentos sobre a estrutura do ambiente (as partes permitidas pela copa Freescale) para limitar ou guiar o trabalho a ser realizado (identificar partes da pista, ao invés de identificar posição absoluta).	
Impacto reavaliado: 3 (médio)	**Probabilidade reavaliada: 4 (médio/alto)**

Elaborado por:	Data:	☐ Respostas incluídas na WBS/Cronograma	Registros adicionais: Verso ou anexos

Formulário sugerido por Gasnier, 2000; Editora IMAN e alterado por Wille.

3 — Análise de Opções Tecnológicas

A análise contida nessa seção tem por objetivo classificar as informações mais relevantes sobre algumas tecnologias disponíveis no mercado e esclarecer quais características foram determinantes na escolha tecnológica para o projeto Robô Mapeador. Nosso projeto é composto por três partes principais: o sistema embarcado, o sistema de comunicação e a estação base. A estação base fornece uma interface para o usuário que permite o envio de comandos de controle de movimento para o robô, além de exibir um traçado da trajetória estimada do carro, baseado nas informações dos sensores. O carro por sua vez envia informações dos sensores para estação base. Este envio é realizado pelo sistema de comunicação, responsável por gerenciar o envio e o recebimento de mensagens.

3.1 Requisitos do Sistema

Nesta seção são apresentados os requisitos do sistema, dividido em três blocos lógicos principais: estação-base, sistema de comunicação e sistema embarcado.

3.1.1 Requisitos da Estação Base

Na tabela 4 estão especificados os requisitos da estação base.

Tabela 4: Tabela com os requisitos e especificações da estação base.

Requisito	**Especificação**
Fornecer uma interface gráfica para o usuário que trace um trajeto.	Software capaz de realizar um traçado 2D e que execute na plataforma PC.
Fornecer uma interface para o usuário enviar comandos.	Interface gráfica com botões para o envio de comandos para o robô.
Apresentar ao usuário os dados dos sensores.	Interpretar os diversos dados dos sensores e oferecer para o usuário a velocidade e posição estimada.

Fonte: Autoria própria

3.1.2 Requisitos do Sistema de Comunicação

Na tabela 5 estão listados os requisitos do sistema de comunicação.

Tabela 5: Tabela com os requisitos e especificações do sistema de comunicação.

Requisito	**Especificação**
Transmitir dados entre o robô e a estação base.	Módulos de comunicação com um alcance mínimo de 20 metros e taxa de transmissão maior que 10Kbps, capazes de operar em modo full-duplex.
Ser compatível com a placa controladora.	Módulos com interfaces de comunicação CAN, I2C ou SPI.

Fonte: Autoria própria

3.1.3 Requisitos do Sistema Embarcado

Na tabela 6 estão presentes os requisistos do sistema embarcado.

Tabela 6: Tabela contendo os requisitos e especificações do sistema embarcado.

Requisito	**Especificação**
Controlar dois motores DC.	Capacidade de gerar ao menos 2 pulsos distintos de PWM com *duty cycle* programável, presença de um driver de motor DC (Ponte-H dupla).
Controlar dois servomotores.	Capacidade de gerar 2 pulsos PWM com frequência de 50Hz e com largura de pulso de 1 a 2 ms.
Fonte energética com autonomia de ao menos 30 minutos.	Bateria de 7,2V e 2000mA.
Capacidade de monitorar a posição, velocidade e orientação do robô.	Capacidade de interação com um encoder de quadratura, um acelerômetro e um giroscópio e com sensores infravemelho.
Capacidade de tratar os dados da câmera de linha.	Presença de um ADC com resolução mínima de 8 bits.
Disponibilidade de componentes em território nacional.	Os sensores, o microcontrolador e o módulo de comunicação sem fio serão comprados em lojas situadas em território nacional.

3.2 Requisitos de Software

3.2.1 Requisitos Funcionais

O Software deverá:

- Mostrar para o usuário a plotagem em três dimensões da pista em tempo real.
- Receber dados dos sensores.

- Realizar o processamento dos dados dos sensores.
- Mostrar as últimas leituras recebidas da câmera.
- Avisar a ocorrência de erros de comunicação.
- Indicar o estado da conexão de comunicação.
- Mostrar uma estimativa de orientação do carro.
- Mostrar uma estimativa da velocidade atual do carro.
- Mostrar uma estimativa da velocidade média do carro.
- Ser capaz de gravar um registro dos principais eventos do sistema.
- Permitir ao usuário desativar o registro dos eventos.
- Permitir ao usuário ativar o registro dos eventos.
- Permitir ao usuário iniciar o mapeamento automático da pista.
- Permitir ao usuário parar o mapeamento automático da pista.
- Permitir ao usuário configurar um número de voltas completas a serem realizadas pelo carro no mapeamento automático.
- Permitir ao usuário enviar comandos para controlar o carro de forma manual.

3.2.2 Requisitos Não-Funcionais

O Software deverá:

- Atualizar a plotagem da pista pelo menos 15 vezes por segundo.
- Permitir ao usuário utilizar teclas de atalho do teclado para fazer o controle manual.
- Impedir o usuário de controlar manualmente o carro se o mapeamento automático estiver iniciado.
- Permitir ao usuário iniciar o mapeamento automático apenas se este não estiver iniciado.
- Permitir ao usuário parar o mapeamento automático apenas se este já estiver iniciado.
- Ser capaz de se recuperar de erros.

3.3 Estação-base

3.3.1 Linguagem de programação

A equipe optou por linguagens de programação mais conhecidas e mais difundidas pela facilidade em encontrar bibliotecas e funcionalidades que sejam úteis para o desenvolvimento do projeto. Dentre essas, também foram descartadas linguagens consideradas inadequadas para os objetivos do projeto, por serem voltadas para outros domínios, como é o caso com Ruby ou PHP, que são voltadas para o desenvolvimento Web.

As linguagens consideradas ficaram, então, sendo C++, Java, C#, Objective-C, Python, Visual Basic e Delphi.

Os critérios adotados para decidir qual das opções seria escolhida envolveram um comparativo entre bibliotecas gratuitas capazes de realizar algumas das tarefas principais necessárias ao desenvolvimento do sistema: comunicação serial, criação de uma interface gráfica e desenho 3D. Foram também consideradas a capacidade de desenvolvimento para múltiplas plataformas, em especial os três sistemas operacionais mais utilizados: Linux (junto com outros sistemas POSIX), Windows e OS X.

Em particular, duas opções foram previstas em relação a C++, uma utilizando as bibliotecas do Qt, desenvolvidas pela Nokia e posteriormente vendidas para a Digia, ou então utilizando outras bibliotecas.

Objective-C, C#, Visual Basic e Delphi foram descartadas por não serem voltadas para o desenvolvimento multi-plataforma.

Tabela 7: Bibliotecas escolhidas para comparação. Foi dada preferência a bibliotecas independentes de plataforma, e indicado caso contrário.

Linguagem	Comunicação Serial	GUI	Desenho 3D
C++	Boost.Asio	FLTK	OpenSceneGraph
C++ utilizando Qt	Suporte nativo	Suporte nativo	Qt3D
Java	RXTX	Suporte nativo	Java3D
Python	pySerial	Tkinter	Pivy

Todas as linguagens comparadas possuem bibliotecas para todas as tarefas e todas as plataformas de desenvolvimento, sendo portanto, segundo esses critérios, igualmente adequadas às necessidades do projeto. Com a venda do Qt da Nokia para a Digia, o formato de licenciamento do SDK foi alterado, e embora as versões antigas sejam disponibilizadas pela GPL e LGPL, as versões mais recentes estão sendo disponibilizadas apenas com uma licensa comercial, o que se mostra uma grande desvantagem em relação às demais, por que o suporte futuro da linguagem é incerto. Devido a estes fatores, como a equipe tem uma experiência maior com a linguagem e as bibliotecas do Java, esta linguagem foi escolhida trabalhar para desenvolver o Software de alto nível na estação-base, de forma a acelerar o processo de desenvolvimento.

3.3.2 Sistema Operacional

Como a linguagem de programação escolhida foi Java, o processo de trabalho se torna independente do sistema operacional, e a estação-base poderá ser executada em qualquer computador capaz de suportar a máquina virtual Java. Para evitar problemas de incompatibilidade entre os ambientes de desenvolvimento, optou-se por utilizar como padrão no projeto de Software o sistema operacional Windows, que a maioria dos integrantes já utiliza, eliminando a necessidade de preparar um ambiente em outro sistema operacional.

3.4 Sistema de Comunicação

O sistema de comunicação tem como objetivo realizar as trocas de mensagens entre o Sistema Embarcado e a Estação Base. A tabela a seguir apresenta alguns sistemas de comunicação utilizados atualmente. Para uso do projeto o bluetooth pode ser decartado por não cumprir um dos requisitos do projeto, já que seu alcance é menor que o mínimo requirido. O sistema Wi-Fi acaba por ter um custo muito elevado inviabilizando a utilização do mesmo. A tecnologia Zigbee acaba por ser mais cara que a RF, o que determinou a escolha desta por possuir um custo mais baixo.

3.4.1 Módulo RF

O Módulo escolhido pela equipe foi o módulo RF nRF24L01 (figura 1) principalmente pelo fato de ser acessível mais rapidamente e ter um custo muito baixo. A equipe encontrou o módulo por uma faixa de preço de 13 reais (sites de compra) até 20 reais (lojas de eletrônica), o que permite diminuir muito o custo do projeto. A escolha do transceptor

Tabela 8: Tabela comparativa dos modulos de comunicação.

	Zigbee (802.15.4)	Wi-Fi (802.11)	Bluetooth	Rádio Frequência
Alcance	30-100 metros	50-300 metros	10 metros	30-100 metros
Taxa Dados	250 Kbits/s	50 Mbits/s	1 Mbit/s	2 Mbits/s
Frequência	2.4 Ghz	2.4 e 5 Ghz	2.4 Ghz	292 Mhz - 2.4 Ghz
Consumo	Baixo	Alto	Médio	Baixo
Custo	Médio	Alto	Baixo	Baixo
Preço Médio	90 Reais	150 Reais	30 Reais	25 Reais

também se deu ao fato de poder ser encontrado em lojas brasileiras e não necessitar de importação.

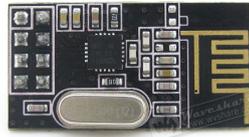



Figura 1: Foto do módulo RF nRF24L01.

Tabela 9: Técnologias de Comunicação

	nRF24L01	RMF12B
Alcance	30 metros	30 metros
Alcance Sem Obstáculos	100 metros	200 metros
Taxa de Transmissão	2 Mbits/s	2 Mbits/s
Alimentação	3.3 V	3.3 V
Consumo de Energia	I_{ativo} = 12,3 mA I_{sleep} = 900nA	I_{ativo} = 25 mA I_{sleep} = 300nA
Frequência	2.4 Ghz	433 Mhz
Dimensões	29 mm X 15 mm	15.9 mm X 16.1 mm
Preço	20 Reais	6.95 Dólares

3.5 Sistema Embarcado

3.5.1 Sensores

Análise da Câmera de Linha

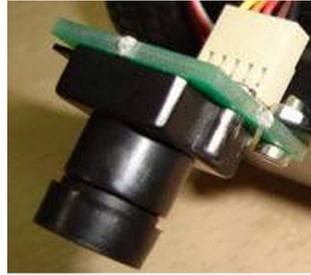
Nesta seção apresentaremosas características técnicas da câmera de linha que será utilizada no projeto como o equipamento principal para a navegação autônoma. A câmera escolhida para o projeto (TSL1401CL) já está inclusa no kit de desenvolvimento da copa Freescale e contém características técnicas que a tornam ideal para executar a tarefa de seguir linha.

Fonte: Adaptado de [3]

Figura 2: Foto da câmera de linha TSL1401CL.

As especificações técnicas principais são as seguintes:

- 128x1 pixels.
- 400 DPI.
- Amplificador de carga interno.
- Alta linearidade e uniformidade.
- Operação em até 8MHz.
- Saída referente à terra.
- Alimentação 3V ou 5V.
- Não necessita de carga resistiva externa.
- Interface externa com apenas 3 pinos.

A principal vantagem desta câmera sobre outras sé o fato de ela realizar a amostragem de apenas uma linha, diminuindo grandemente a quantidade de processamento que o microcontrolador deverá efetuar, um fator crítico para o projeto. Outra vantagem é a escala com referência a terra, o que facilita a interface com o ADC do microcontrolador e a quantidade de pinos utilizada para realizar a interface, apenas 2 GPIO e um ADC.

A interface com o microcontrolador ocorre por meio de 3 pinos presentes na câmera, cujo encapsulamento pode ser visualizado na figura 3. Os pinos são CLK (clock), SI (serial input) e AO (analog output). O clock determina a taxa de envio da leitura dos pixels para o microcontrolador, e é limitado em 8MHz, além de realizar um latch do sinal SI. O SI (serial input) determina o momento em que a câmera iniciará a exposição dos pixels à luz e por fim o AO transmite a tensão analógica (0 a V_{dd}) de cada fotodiodo presente na câmera. Para a realização do interfaceamento com a câmera com o microcontrolador só necessitamos de 2 GPIO e um ADC.

Conectado a cada fotodiodo presente na câmera existe um circuito integrador, que integra a corrente gerada pela exposição luminosa no fotodiodo. Desta forma, podemos controlar a quantidade de exposição de luz que a câmera recebe antes de ter os valores analógicos amostrados, permitindo um ajuste fino da sensibilidade à luminosidade. O tempo de integração é o tempo entre dois pulsos SI e pode ser controlado pelo usuário.

Giroscópio

Um giroscópio é um instrumento que consiste de um rotor suspenso por um suporte formado por dois círculos articulados. Quando o dispositivo entra em rotação, tende a se opor às mudanças de direção do próprio dispositivo, captando a variação da posição do eixo e analisando se o eixo mudou de direção ou não.

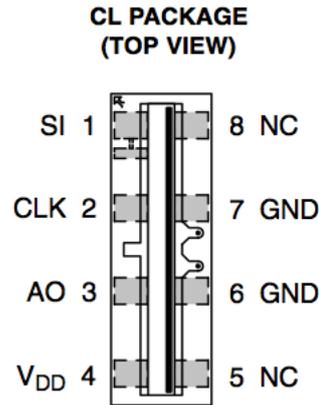
O giroscópio funciona seguindo um fenômeno denominado precessão. A precessão ocorre quando um objeto que está em rotação com um eixo, tentará girar perpendicular a esse

Fonte: Adaptado de [3]

Figura 3: Encapsulamento da câmera de linha TSL1401CL.

eixo (ou a 90 graus desse eixo). A figura 4 ilustra melhor o funcionamento de um giroscópio, quando a roda de uma bicicleta gira, ela gira perpendicularmente ao eixo de rotação. O giroscópio funciona com 3 desses eixos movendo livremente e se opondo a qualquer tentativa de mudar a direção da rotação [1].

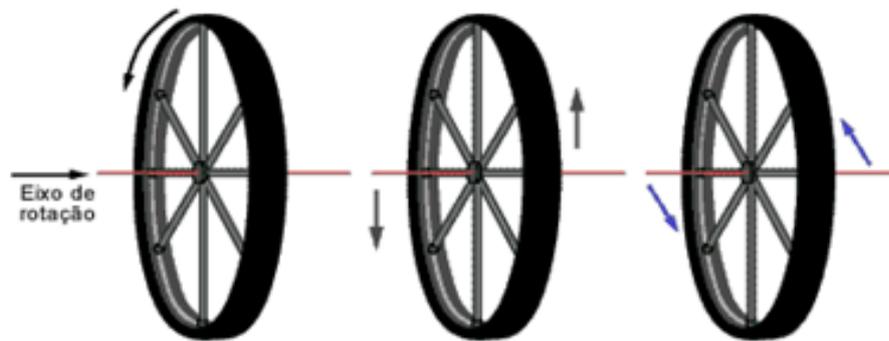



Figura 4: Fenômeno de precessão [1]

A escolha do giroscópio que será utilizado no projeto se deu ao fato de acessibilidade do mesmo. O módulo MPU-6050 (figura 5) foi emprestado por um dos professores da disciplina, assim diminuindo o custo do projeto. Também se deu ao fato de possuir um acelerômetro integrado, sensor que também será utilizado no projeto. O módulo possui saída I2C e será acoplado ao microcontrolador por meio de uma placa PCB.

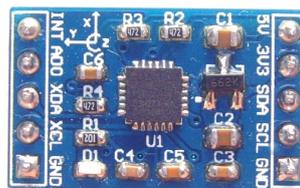



Figura 5: Foto do Giroscópio e Acelerômetro MPU-6050.

Inicialmente, o grupo decidiu usar uma bússola para captar a variação de direção, mas a bússola elevaria mais o custo do projeto, visto que estas acabam por ser mais caras que o giroscópio utilizado. As bússolas mais baratas não atenderiam aos requisitos do projeto

pois possuem uma resolução menor que a pretendida para captar a variação de direção do carrinho. As especificações técnicas do giroscópio (MPU-6050) estão listadas na tabela 10 e da bússola (CMPS03) na tabela 11.

Tabela 10: Características do Giroscópio

Características	MPU-6050
Eixos	3 eixos
Escala graus/segundo	±250, ±500, ±1000, ±2000
Tensão	3.46 V
Corrente Máx.	3.9 mA
Corrente Típica	3.6 mA
Low Power	5 μA
Dimensões	22.1 mm X 13.6 mm
Preço	15 Dólares

Tabela 11: Características da Bússola

Características	CMPS03
Resolução	0.1 grau
Tensão	5 V
Corrente	20 mA
Dimensões	32mm X 35 mm
Preço	45 Dólares

Acelerômetro

Os acelerômetros são dispositivos eletromecânicos que relacionam a aceleração com um sinal de saída, assim sendo possível medir a aceleração sobre objetos. Este sinal de saída pode ser analógico ou digital. Os modelos de acelerômetros são capazes de detectar a magnitude e o sentido da aceleração e podem ser interpretadas por outros dispositivos.

O acelerômetro tem como objetivo captar a variação de plano do carrinho. Logo, quando o carrinho subir uma rampa, o sensor poderá reconhecer a rampa para mapear a mesma. Para o projeto, foi escolhido o módulo MPU 6050, que contém um acelerômetro, pricipalmente por fácil disponibilidade (fornecido pelos professores da disciplina).

Os acelerômetros capacitivos (utilizados nesse projeto) seguem o princípio da Segunda Lei de Newton onde existe uma massa presa a uma mola fixada entre duas placas paralelas. A aceleração da massa causa variação da distância devido ao movimento livre da placa e, portanto resulta em uma mudança na capacitância elétrica [4].

Um acelerômetro também pode ser capaz de medir a aceleração de um objeto, além de conseguir medir choque e vibrações.

O módulo possui saída I2C e para o projeto uma escala de 2g atenderá aos requisitos do projeto. Suas características podem ser observadas na tabela 12.

Protocolo I^2C

O protocolo I^2C é um sistema de comunicação desenvolvido pela Philips e amplamente utilizado em dispositivos eletrônicos. A comunicação ocorre entre dois dispositivos,

Tabela 12: Características do Acelerômetro

Características	MPU 6050
Eixos	3 eixos
Escala	±2g a ±16g
Tensão	3.46 V
Corrente Máx.	3.9 mA
Corrente Aceleromêtro	500μA
Low Power	10 μA
Dimensões	22.1 mm X 13.6 mm

sendo um deles denominado Master (ou mestre) e o outro denominado Slave (ou escravo). O Mestre geralmente e o dispositivo com maior capacidade de processamento.

Em um sinal, os dados são lidos e recebidos pela forma de bits (representação de número binário), onde o primeiro pulso de sinal contem a intenção do controlador de ler ou enviar dados ao respectivo dispositivo.

O protocolo segue basicamente duas etapas. Primeiramente o dispositivo mestre sincroniza os dois dispositivos enviando um sinal de clock (SCL). Posteriormente ocorre a transmissão dos dados entre eles pela via do serial data (SDA).

Segundo Dias Filho [5] a comunicação entre os dois dispositivos e a transmissão dos dados e dada por uma sequencia de oito bits e ocorre da seguinte maneira:

1. O dispositivo Master ajusta as condições iniciais. 2. O dispositivo Master envia sete bits para o Slave para dizer com qual dispositivo se deseja comunicar (envia o endereço do dispositivo). 3. O Master envia o oitavo bit dizendo se deseja ler ou escrever no dispositivo. 4. O dispositivo Slave retorna um bit para o Master dizendo que entendeu a comunicação (Acknowledge bit). 5. O dispositivo Master ou Slave envia pacotes de oito bits contendo a informação desejada sempre seguido de um ACK (Acknowledge bit) dizendo se o Slave ou Master recebeu as informações. 6. A comunicação termina quando o Master envia um bit de parada (ou stop bit).

Um exemplo do protocolo de comunicação I^{2}C pode ser visto na figura 6, onde SDA é o barramento de dados, servindo para os dois lados (comunicação entre o master e o slave ou comunicação entre o slave e o máster) e SCL é o clock da comunicação.

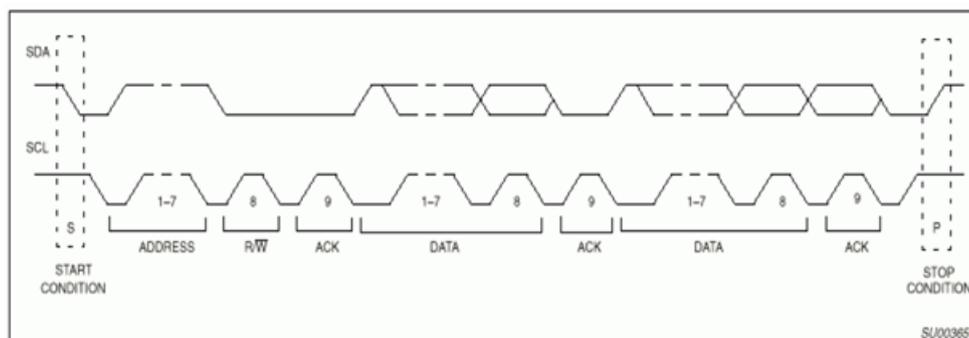



Figura 6: Protocolo de Comunicação I^{2}C [5].

Encoder

Para contar o número de rotações de cada motor será utilizado um encoder. Um encoder é um dispositivo que lê o movimento e transforma em informação. Existem diversos tipos de encoder, entre eles o que utiliza a reflexão de sinais. Para isso ocorrer um disco com partes transparentes e opacas é colocado no motor, ao passar pela parte transparente o sinal enviado é recebido pelo outro sensor e ao passar na parte opaca e sinal é bloqueado. Assim é possível saber por quantos setores o motor girou e dado que o tamanho do disco é conhecido podemos saber a rotação do motor.

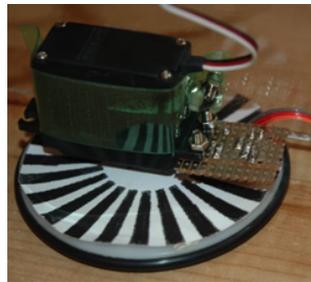
Para o nosso problema em específico, esse é um tipo de encoder se encaixa bem, já que poderíamos utilizar um encoder dentro da estrutura do carrinho (acoplando o encoder na roda), porém devido a falta de espaço no kit essa opção se tornou inviável. A outra opção seria de colocar na parte externa da roda o disco e fixar um sensor para medir a rotação. Na figura 7 podemos ver um exemplo de encoder.

Fonte: Adaptado de [3]

Figura 7: Foto de um encoder.

Primeiramente, para acoplar o encoder fora do carrinho e ler os dados gerados pela roda, o grupo pretendia acoplar o encoder na roda do carro. Para isso, seria necessário acoplar um *stripe code wheel* (disco composto por listras pretas e branca) na roda e o encoder na parte traseira do carro, movendo o sensor fototransmissor para ler o disco (que está acoplado a roda), assim o encoder poderia ler quantas rotações a roda realizou por minuto e medir a velocidade.

O encoder RB-Sbo-33, figura 8, consegue atender aos requisitos do projeto, suas especificações podem ser visualizadas na tabela 13.

Tabela 13: Tabela com as especificações do encoder.

Características	Wheel Watcher Rotation Encoder
Clocks por rotação	128
Tensão	4.5 V - 5.5 V
Corrente Máx.	30 mA
Largura pulso de clock	25 μs
Dimensões	5.5 cm X 2.3 cm

Também é possível montar um encoder utilizando fototransistores, acoplando o *stripe code wheel* na roda do carrinho e lendo os pulsos de clock gerado pelo fototransistor.

O grupo decidiu descartar essa opção. o encoder iria ter problemas para ler o *stripe code wheel*, pois o disco iria permanecer fixo (na roda), enquanto o encoder (que está acoplado

Figura 8: Foto de um encoder RB-Sbo33.

a parte traseira do carro) iria se movimentar, devido a suspensão do carro, prejudicando a leitura do mesmo.

Outra opção, seria utilizar um encoder na engrenagem do motor da roda, assim o encoder não teria problemas com a leitura. Um modelo de *stripe code wheel*, que pode ser observado na figura 9 ao lado do encoder, seria acoplado na engrenagem enquanto o encoder ao lado do círculo. O modelo de ligação pode ser observado na figura 10 onde é possível observar o encoder e o círculo acoplados ao carro.

Figura 9: Encoder fotoelétrico utilizado no carro.

O encoder apresentado na figura 10 é compatível com o carrinho e é demonstrado na tabela 14.

Tabela 14: Tabela com as especificações do encoder utilizado no carro.

Características	Encoder fotoelétrico
Número de linhas (disco)	100 linhas
Tensão	5 V
Distância interna do encoder	3.5 mm
Dimensões	22 mm X 7.5 mm

Figura 10: Encoder acoplado ao carro.

O uso deste encoder se mostrou problemático ao longo do projeto e foi substituído por um sensor de efeito hall, que será explicado posteriormente.

Sensor de efeito hall

Outra opção para medir a velocidade do carro seria a utilização de um sensor de efeito hall. Seria necessário acoplar pequenos imãs nas rodas para ativar o sensor. Essa opção se torna necessária caso não seja possível acoplar o encoder o carro, devido a impossibilidade física. No sinal da saída do sensor há uma variação na saída quando um imã está próximo. Para utilizar esse sensor como um medidor de velocidade imãs são colocados na roda e e cada vez que se aproxima ativa o sensor. Uma desvantagem deste método seria a perda de precisão. Enquanto um disco de um encoder pode ter até 100 ranhuras, seria possível colocar apenas 4 imãs na roda. O que gera uma grande perda de precisão. Na figura 11 temos um sensor de efeito hall.

Figura 11: Sensor de efeito hall US1881

3.5.2 Servomotor

Na execução deste projeto será necessário um servomotor para controlar a direção das rodas dianteiras do robô. O servomotor já está incluso no kit de desenvolvimento Freescale, Futaba S3010 Standard High-Torque, e supre as necessidades do projeto. Este servomotor pode ser visualizado na figura 12.

Figura 12: Foto do servomotor Futaba S3010.

3.5.3 Microcontrolador

Para seguir as normas da Freescale, tinhamos duas opções de placas de desenvolvimento a TRK-MPC5604B (figura 13), com o microcontrolador MPC-5604B e a placa FRDM-KL25Z (figura 14), equipada com o microcontrolador KL25Z128VLK4. Neste projeto necessitamos de ao menos um módulo I^2C, para controlar o acelerômetro/giroscópio; um módulo SPI, para os transdutores *Wireless*; Também precisamos de 4 canais PWM para controlar os servomotores e os motores DC. A câmera de linha por sua vez necessita de um sinal de clock e 2 GPIOs enquanto a placa de LEDs requer apenas 8 GPIOs para a leitura. Como ambas as placas satisfazem nossos requisitos de conectividade e performance mas só encontramos kits de desenvolvimento com a segunda placa, esta foi a opção adotada pela equipe. Na tabela 15 podemos observar uma tabela comparativa entre as duas plataformas. A plataforma escolhida utiliza um microcontrolador com arquitetura ARM, opera em 32-bits e foi desenvolvido sobre o CortexTM-M0+ core. Um diferencial da segunda placa é que a distância entre os pinos e o layout é compatível com o padrão Arduino Revision 3 (R3), fazendo com que seja possível instalar qualquer placa periférica compatível com o Arduino diretamente.

Tabela 15: Tabela comparativa das plataformas disponíveis.

Característica	TRK-MPC5604B	FRDM-KL25Z
Barramento	32 bits	32 bits
Frequência Máx. de Clock	64 Mhz	48MHz
Resolução do Módulo ADC	10 bits	16 bits
SPI	3	2
I^2C	1	2
Canais PWM	56	10
GPIO	79	20
Suporte USB	Sim	Sim
Debug	JTAG	OPENSDA
SRAM	48 Kb	16 Kb
Flash	64 Kb	128 Kb

Figura 13: Foto da placa de desenvolvimento MPC5604B.

Figura 14: Foto da placa de desenvolvimento FRDM-KL25Z.

3.5.4 Análise do Chassi

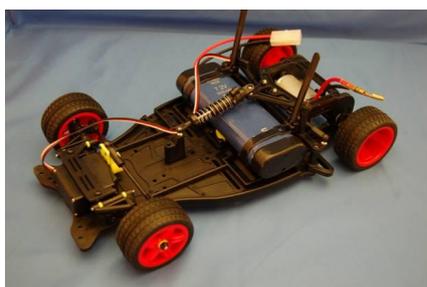
O chassi utilizado para a montagem do robô será o contido no kit da Freescale, pois o seu uso inalterado é uma regra da competição. As únicas modificações permitidas são a realização de furos e a anexação de peças de suporte auxiliares, contanto que o tamanho total se mantenha dentro do limite máximo (25,0 cm de largura, 40,0 cm de comprimento e 30,5 cm de altura). Uma modificação no chassi se faz necessária para a anexação do encoder , respeitando os limites de tamanho para a competição. Na figura 15 pode-se visualizar o chassi já montado.

Fonte: Adaptado de [2]

Figura 15: Foto do chassi.

4 — Arquitetura geral do HW e do SW

Para termos uma visão geral do projeto do robô mapeador, podemos dividi-lo em três grandes blocos distintos: a estação-base, o sistema de comunicação e o robô. Estes blocos podem ser visualizados na figura 16.

Figura 16: Diagrama do sistema em alto nível.

A estação-base é a responsável pela interface com o usuário e é constituída fisicamente por um computador, que recebe os comandos do usuário por meio do mouse e teclado, por um arduíno e um módulo de rádio-frequência. A interpretação destes comandos é realizada por meio de um software em Java, responsável por processar as informações, montar um pacote contendo um comando ou configuração, enviar para o arduino da base e também processar o recebimento de um pacote.

O arduino tem como função receber os pacotes contendo dados dos sensores oriundos do robô através do módulo wireless e de transmitir pacotes recebidos do software de controle da estação base para o módulo, para que sejam enviados ao robô. A comunicação entre o computador e o arduino é realizada através de uma interface serial, e é implementada no computador por meio da biblioteca RxTx da linguagem Java. A comunicação entre o arduino e o módulo, por sua vez, é realizada através do protocolo SPI.

Podemos visualizar a estrutura da estação base na figura 17.

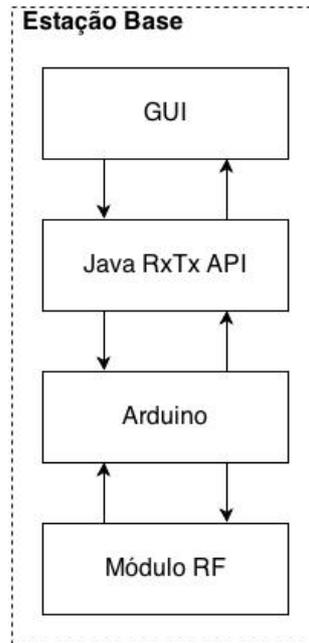



Figura 17: Diagrama estrutural da estação base.

Como pode ser visualizado na figura 18, o centro controlador do robô é a placa Freescale Freedom Board. Ela é responsável por tratar os dados da câmera de linha, implementar o algoritmo de controle, controlar o driver dos motores DC e controlar a direção do servomotor do robô, para realizar curvas. Também está conectada a um módulo RF, que se comunica de forma unidirecional com da estação base e é capaz de receber comandos vindos da base. O envio dos dados dos sensores de efeito hall para a estação base é realizado através de um Arduino, que contabiliza os pulsos e transmite de forma unidirecional e contínua para a estação base.

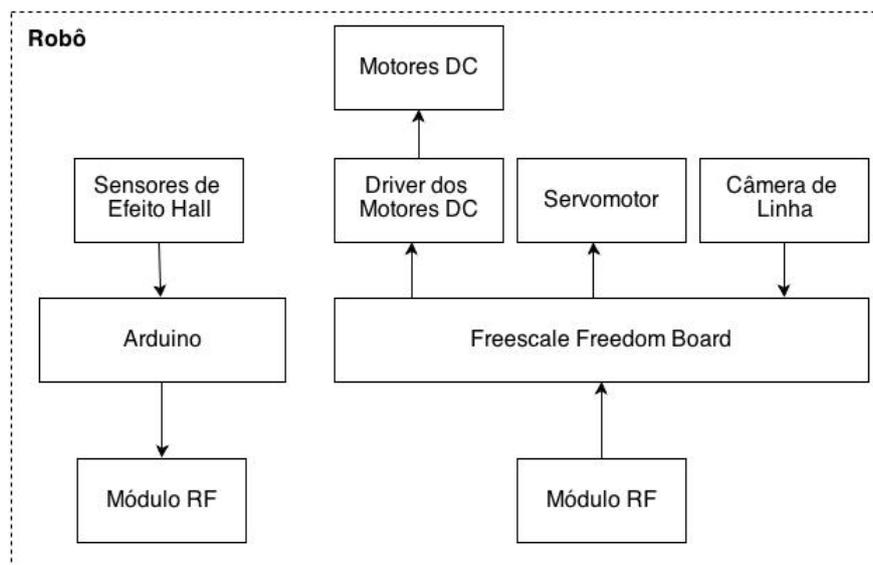



Figura 18: Diagrama estrutural do robô.

5 — Estação-base

5.1 Diagrama de Sequência

O diagrama de sequência, visualizado no apêndice C e presente no cd que contém a documentação, indica todas as ações feitas pelo usuário e pelo robô e a sequência de cada uma das funções e em qual ordem elas são acionadas dentro do software.

Primeiramente, é permitido ao usuário atualizar uma porta serial ou conectar em uma porta serial. Quando o usuário clica no botão de atualizar as portas seriais, o software da estação base ativará a função serialSerachButtonActionPerformed para identificar que o botão foi acionado, a função findPorts para procurar no computador portas seriais disponíveis e a função serialComboBox.add para listar as portas seriais disponíveis. Quando o usuário clica no Botão do conectar com a porta serial, o software de estação base ativará a função connect para iniciar a conexão com a porta serial selecionada. Tendo conexão com a porta serial realizada, os botões de controle sobre o robô são habilitados pela função enableAll.

Com a porta serial estabelecida, o software adiciona a classe SerialPort a classe Parser, que é responsável por receber informações via serial, e também inicializa a classe Messenger, que tem como objetivo enviar informações via serial.

O software terá um slider para controlar o PWM do robô, sendo um slider para a movimentação automática e outro para o modo manual. Ao deslizar o slider de movimentação automática, é ativada a função autoPWMSliderStateChanged para identificar o deslizamento e a função autoPWMTextBox.setText para alterar o texto do campo, indicando o valor do PWM no modo automático. Deslizando o slider de movimentação manual, é ativada a função manualPWMSliderStateChanged para identificar o deslizamento e a função manualPWMTextBox.setText para alterar o texto do campo indicando o valor do PWM no modo manual.

Para a movimentação do robô no modo manual são acionadas as funções do software referentes a cada botão. As funções de actionPerformed (frontButtonActionPerformed, rightButtonActionPerformed, leftButtonActionPerformed e backButtonActionPerformed) geram o evento para movimentar o robô para frente, direita, esquerda e trás respectivamente. Após o evento ser gerado, função manualCommand envia a classe Messenger todas as configurações referentes a ação do robô (como ângulo, valor do PWM entre outros) e a função sendMessage envia via serial a mensagem formada, mensagem esta que será enviada diretamente ao robô.

Quando o usuário clica no botão de habilitar movimentação automática, o software ativa a função EnableManualMovement para desabilitar o modo manual (passando como parâmetro um valor falso para a função, assim desabilitando ela) e através da função automaticMode (que passa como parâmetro um valor verdadeiro, já que é desejado ativar o modo automático) passa para a classe Messenger a informação de habilitar movimentação automática (seguindo o mesmo caminho dos comandos manuais explicados acima). Na classe Messenger é ativada a função SendMessage que envia o comando ao robô.

Quando o usuário clica no botão de desabilitas movimentação automática, o software ativa a função EnableManualMovement para habilita-la, ativando o modo manual do robô. A seguir, a função automaticMode é ativada (e passa como parâmetro um valor falso, já que é desejado desabilitar essa função) e envia para a classe Messenger a informação de desabilitar a movimentação automática. A seguir, a classe Messenger envia a mensagem para o carro através da função SendMessage.

No momento que o usuário fecha a janela do software da estação de base, o software

deverá fechar a conexão com a porta serial através da função disconnect, liberar esta porta serial removendo o listener pela função serialPort.removeEventListener e fechar a comunicação com a classe Messenger.

5.2 Interface Gráfica do Software da Estação Base

Para este projeto, foi necessário criar um software para a estação base para que o usuário tenha acesso à informações sobre o trajeto e possa movimentar o robô. O software da estação base será implementado em JAVA e terá sua interação com o usuário através de uma interface gráfica. A interface gráfica do usuário possui o design mostrado na figura 19.

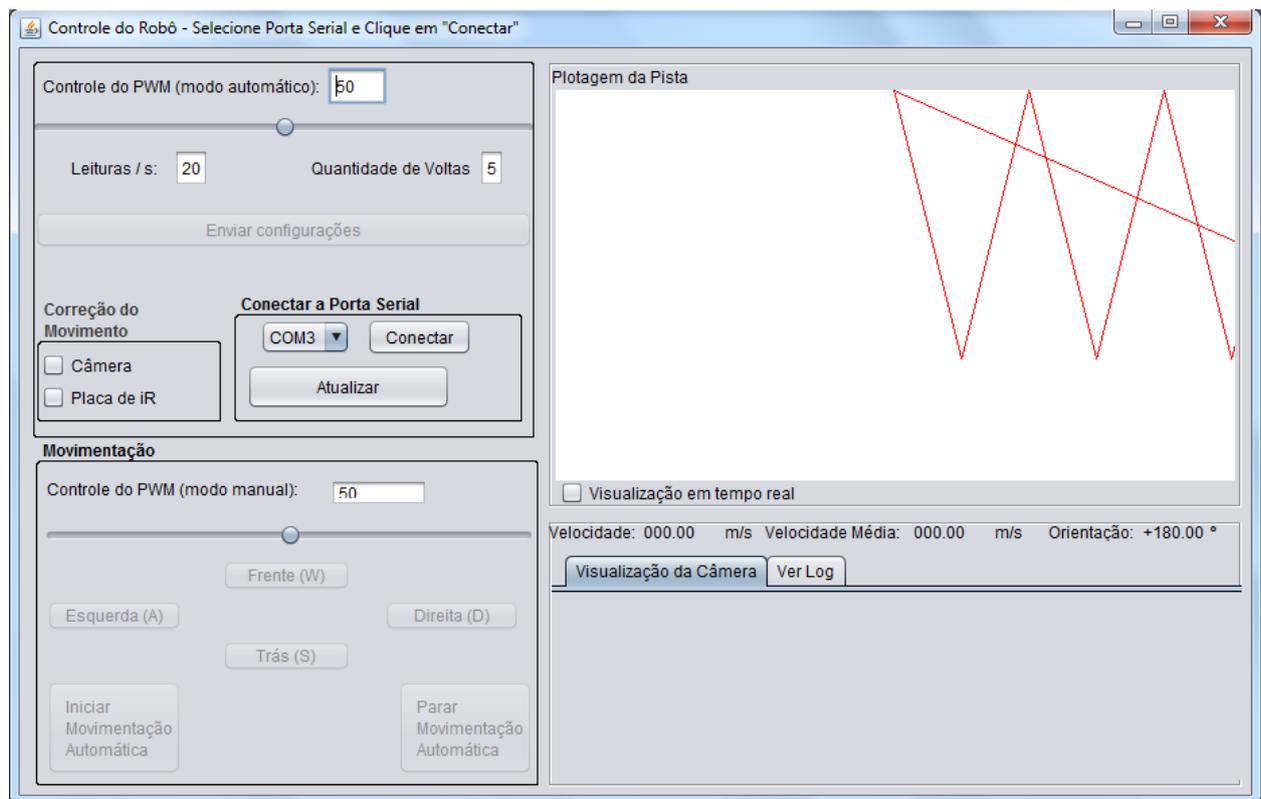



Figura 19: Visualização da tela do software da estação base

No painel localizado ao canto superior esquerdo, temos o painel para configurações da movimentação automática do robô e de conexão com a porta serial. Dentre as configurações de movimentação automática, temos o PWM, a quantidade de leituras por segundo e a quantidade de voltas a serem dadas na pista. Ao fazer uma alteração nas configurações, o software enviará o comando de alteração somente quando o usuário clicar no botão "Enviar configurações". No campo de texto à direita do texto "Controle do PWM (modo automático)", o usuário poderá visualizar o valor do PWM utilizado pelo robô para controlar a velocidade de rotação das rodas. Para alterar o valor do PWM é necessário deslizar o slider encontrado logo abaixo do texto "Controle do PWM (modo automático)".

Para configurar a quantidade de leituras feitas por segundo dos sensores, o usuário deverá digitar a quantidade desejada no campo de texto localizado à direita do texto "Leituras/s". Para alterar a quantidade de voltas a serem feitas, o usuário deverá digitar o valor desejado no campo à esquerda do texto "Quantidade de Voltas".

Para configurar a forma que a correção da movimentação, o usuário deverá selecionar as caixas de seleção correspondentes aos sensores desejados dentro do campo "Correção do

Movimento". Para selecionar a correção pela câmera, o usuário deverá selecionar a caixa à esquerda do texto "Câmera". Para selecionar a correção pela placa de infravermelho, o usuário deverá selecionar a caixa à esquerda do texto "Placa de iR".

No subpainel "Conectar a Porta Serial", o usuário deverá selecionar a qual porta serial o software se comunicará para receber os dados do módulo de RF. Para atualizar a lista de portas seriais (caso a porta serial desejada não esteja aparecendo), o usuário poderá clicar no botão "Atualizar". Para conectar à porta serial, o usuário deverá selecionar a porta serial desejada na lista (combo Box) e clicar no botão "Conectar".

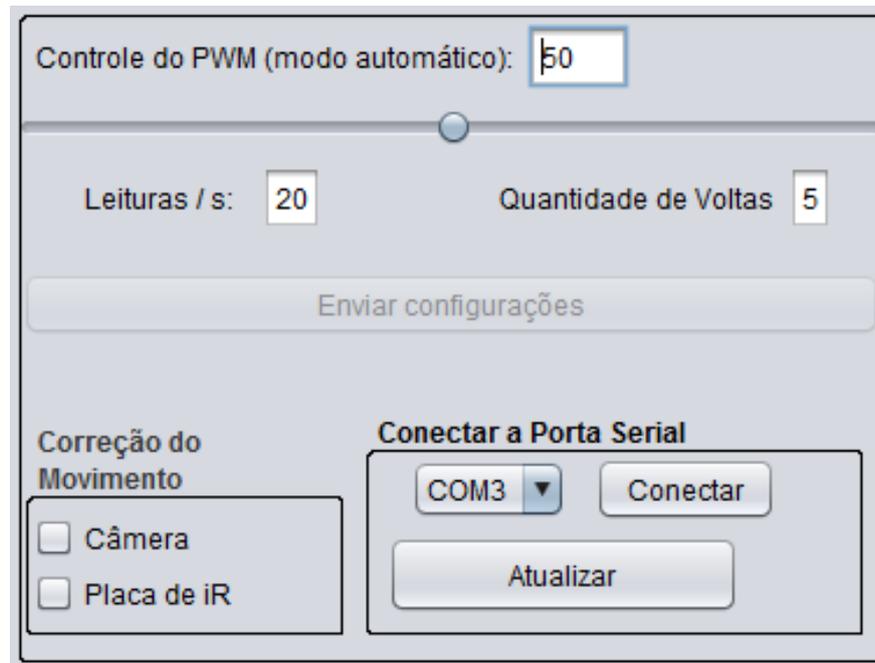



Figura 20: Painel de Controle da Movimentação Automática e de conexão com a Porta Serial

No canto inferior esquerdo, temos o quadro de movimentação do robô, tendo os botões para direcionar a movimentação do robô e um botão para habilitar movimentação manual (movimentação controlada pelo usuário) e outro para desabilitar a movimentação manual. Neste painel, também é possível controlar o PWM do robô, alterando a velocidade de rotação das rodas do robô.

Para habilitar a movimentação do robô por comando do usuário, o usuário deverá antes habilitar a movimentação manual clicando no botão "Parar Movimentação Automática", clicando neste botão, os botões de movimentação e o botão "Iniciar Movimentação Automática" ficarão habilitados e o botão "Parar Movimentação Automática" ficará desabilitado. Clicando no botão "Iniciar Movimentação Automática", os botões de movimentação manual e o botão "Iniciar Movimentação Automática" ficarão desabilitados e o botão "Parar Movimentação Automática" ficará habilitado, esta ação envia um comando ao robô de iniciar movimentação automática.

O campo de texto à esquerda do texto "Controle do PWM (modo manual)" mostra o valor do PWM, o qual pode ser alterado pelo slider encontrado logo abaixo. Para movimentar o robô para frente, o usuário deverá clicar no botão "Frente (W)", para a direita clicando no botão "Direita (D)", para a esquerda clicando no botão "Esquerda (A)", e para trás clicando no botão "Trás (S)".

No canto superior direito, temos o painel de plotagem da pista. A pista será desenhada na área branca com uma linha vermelha baseada nos pontos calculados à partir de leituras dos sensores. A caixa de seleção no canto inferior esquerdo, seguido do texto "Visu-

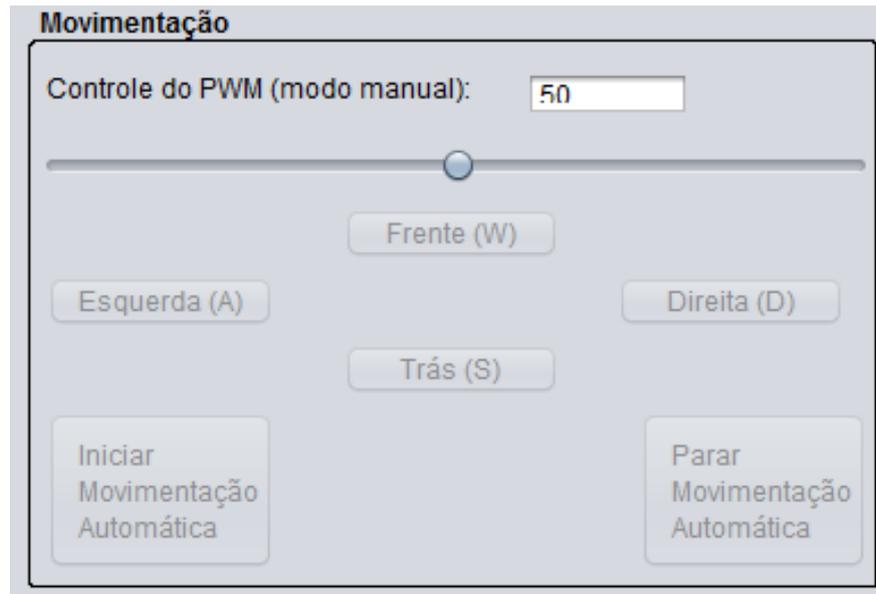



Figura 21: Painel de Controle de movimentação manual do robô

alização em tempo real" ativa ou desativa a visualização em tempo real. Se a visualização em tempo real estiver desabilitada, a pista não será desenhada em tempo real.

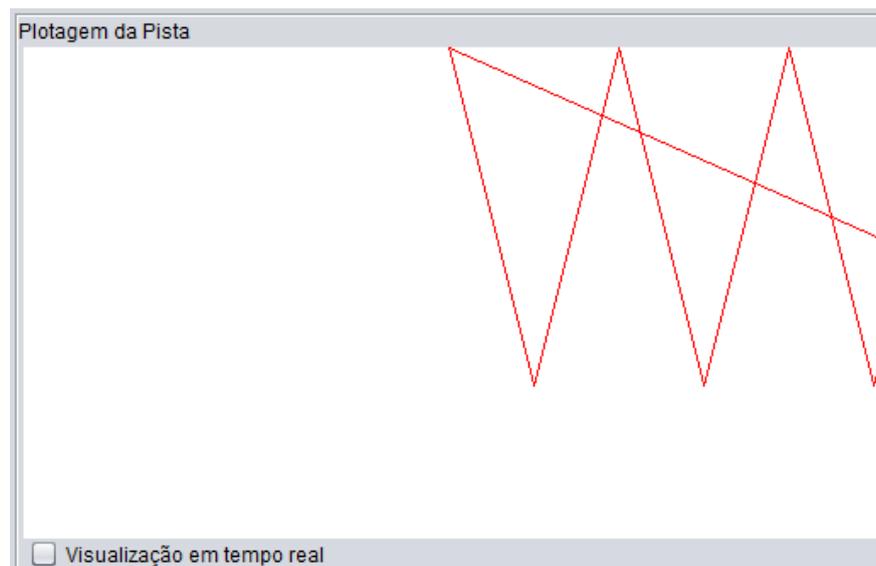



Figura 22: Painel de Plotagem da pista

No canto inferior direito, podemos visualizar o quadro de informações para o trajeto atual, como velocidade atual, velocidade média, orientação do robô em relação à posição inicial, a visualização e a visualização do Log.

À direita do texto "Velocidade: " temos a velocidade atual do robô dada em metros por segundo, à direita disto, temos um campo "Velocidade Média: " seguido da velocidade média atual, dada em metros por segundo, à direita disto e como última informação, temos o texto "Orientação: " seguido de um sinal e um valor numérico, o sinal "+" indica rotação de um valor em graus no sentido horário e o sinal "-" indica rotação de um valor em graus no sentido anti-horário.

Neste painel, podemos alternar entre a visualização da imagem da câmera de linha clicando na aba "Visualização da Câmera" ou ver os registros das atividades realizadas na

porta serial na aba “Ver Log”.

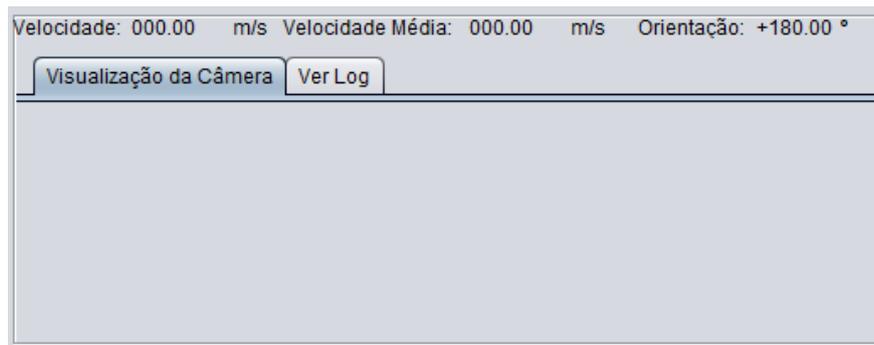



Figura 23: Painel de visualização de informações do trajeto atual, exibindo a aba de visualização da imagem da câmera de linha

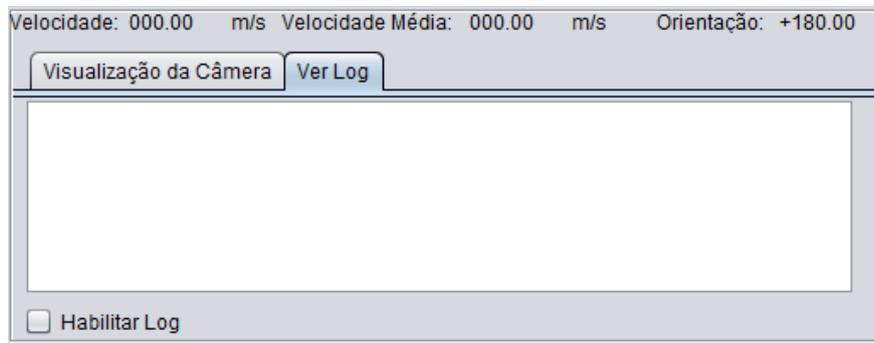



Figura 24: Painel de visualização de informações do trajeto atual, exibindo a aba de visualização do log

6 — Sistema de comunicação

Foi projetado um protocolo de comunicação entre os módulos nRF24L01 utilizando um payload estático de 32 bytes. Foram tomadas medidas para garantir, caso não haja falhas de comunicação, o recebimento do payload, utilizando um protocolo para a recuperação de erros. Também foi projetado o formato das mensagens.

6.1 Troca de Mensagens

Tomando como base o protocolo Enhanced ShockBurst, do módulo nRF24L01, foi projetado um protocolo de troca de mensagens onde, a princípio, a estação base faria o controle total da comunicação. Para isso, a estação base sempre inicia a comunicação, enviando um comando ou enviando uma requisição caso deseje receber dados do Sistema Embarcado. Assim, o carro iria permanecer no estado de recebimento de pacotes e caso recebesse um comando para enviar dados dos sensores para a Estação Base (comando este enviado pela Estação Base), ele passaria para o estado de envio.

Porém o grupo teve dificuldades no desenvolvimento da comunicação. Sendo que foi necessário adionar outro microcontrolador para enviar os dados dos sensores para a Estação Base (tópico este que será discutido adiante). O protocolo foi alterado para incluir outro microcontrolador (Arduino que foi acoplado ao carro junto com a placa KL25Z), que servirá somente para enviar dados dos sensores. A placa KL25Z continuou sendo utilizada para controlar o carro. O protocolo funciona da seguinte maneira:

1) A Estação Base pode enviar um comando para a placa KL25Z ou receber um valor do Arduino.

2) A placa KL25Z permanece somente no modo de receber comando não enviando nada concreto a Estação Base (somente um ACK).

3) O Arduino espera uma mensagem de sincronização e, após recebe-la, envia continuamente dados dos sensores para a Estação Base.

6.2 Formato das mensagens

Foi estabelecido um formato geral para as mensagens, onde o primeiro byte identifica o tipo de mensagem, o segundo pode conter flags de configuração ou subtipo, e um formato diferenciado para os 30 bytes restantes, como pode ser visto na tabela 16.

Tabela 16: Formato geral de mensagens

Tipo	Flags	Mensagem
1 byte	1 byte	30 bytes

Em seguinda serão mostrados os tipos e o formato de cada tipo de mensagem.

6.2.1 Envio de leitura

Esse tipo de mensagem envia leituras dos sensores para a estação base.

O formato do pacote é composto seguidamente por dois bytes que representa a leitura do ângulo do giroscópio e um byte com a leitura do sensor de efeito hall. Cada byte

da mensagem possui o valor da leitura dos sensores no instante de 50ms sendo os próximos bytes completados no próximo instante de 50ms (por exemplo, no instante de 50ms é pego o primeiro valor dos sensores e colocado nos três primeiros bytes, no próximo instante de 50ms é pego outros valores dos sensores e colocado nos próximos três bytes até a mensagem se completar). O último byte do pacote contém a flag que representa o tipo de mensagem. Uma exemplificação do pacote pode ser vista na figura 25.

Ângulo giro	Hall	Ângulo giro	Hall	Ângulo giro	Hall	...	0x01
2 bytes	1 byte	2 bytes	1 byte	2 bytes	1 byte		1 byte

Figura 25: Mensagem de envio de leituras.

6.2.2 Passagem pela linha de chegada

Esse tipo de mensagem é enviada pelo sistema embarcado para avisar à estação base que ele passou pela linha de chegada.

Tabela 17: Mensagem de passagem pela linha de chegada

0x02	Não utilizado Não utilizado
1 byte	31 bytes

6.2.3 Comandos manuais

Esse tipo de mensagem é enviado da estação base para o sistema embarcado para fazer o controle manual do carro.

Tabela 18: Mensagem de envio de leituras

0x04	Tipo de comando	PWM	Ângulo	Não utilizado
1 byte	1 byte bitmask	4 bytes	1 bytes	25 bytes

O segundo byte indica qual tipo de movimento o carro deve fazer (apenas um pode estar ativo por vez):

- 0x01: iniciar movimento para a frente
- 0x02: iniciar movimento para trás
- 0x04: virar rodas para a direita
- 0x08: virar rodas para a esquerda
- 0x10: parar
- 0x20: centralizar roda

O resto da mensagem indica a intensidade do movimento.

- PWM: Número de 0 a 1, indicando a fração de tempo ativo do PWM, para controlar a velocidade do carro
- Ângulo: Número inteiro, indicando o ângulo de deslocamento desejado

Tabela 19: Mensagem de configuração do sistema embarcado

0x08	Flags de configuração	Ângulo	Clock	PWM	Leituras/s	Uso dos sensores de controle	Não utilizado
1 byte	1 byte bitmask	1 byte	1 byte	4 bytes	1 byte	1 byte	22 bytes

6.2.4 Configuração do sistema embarcado

Esse tipo de mensagem é enviado da estação base para o sistema embarcado, para configurar várias opções.

O restante da mensagem passa os parâmetros de configuração:

- Ângulo: Posição desejado do servomotor da câmera.
- Clock: Frequência de operação desejada para a câmera.
- PWM: Número de 0 a 1, indicandso a fração de tempo ativo do PWM, para controlar a velocidade do carro
- Leituras: Número de leituras a serem realizadas e enviadas por segundo
- Controle: 0x01 para usar apenas a câmera, 0x02 para usar apenas o sensor infravermelho e 0x03 para utilizar os dois.

6.2.5 Mapeamento automático

Esse tipo de mensagens é enviado da estação base para o carro para controlar o início e o fim do mapeamento automático.

Tabela 20: Mensagem de controle do mapeamento automático

Tipo	Subtipo de mensagem	Número de voltas	Não utilizado
1 byte	1 byte	1 byte	30 bytes

Subtipos de mensagem:

- 0x01: Inicia o mapeamento automático
- 0x02: Encerra o mapeamento automático
- 0x04: Modiica o número de voltas a serem realizadas

O campo Número de voltas indica quantas voltas ainda deverão ser realizadas pelo carro antes de parar, e não é válido se o comando for o de encerrar o mapeamento. Caso seja passado o valor 0, o carrinho não irá parar a não ser que receba o comando de encerrar o mapeamento.

6.3 Diagramas de Comunicação

A figura 26 apresenta os possíveis estados que o carro pode assumir durante a sua execução. Primeiramente todos os pinos de entrada e saída, PWM, módulo RF e a câmera são inicializados. O carro passa para o modo manual e aguarda uma instrução vinda da Estação-Base. Utilizando o primeiro byte da mensagem, o carro decodifica a instrução recebida e executa essa instrução.

Modo Controle Automático: O carro passa para o modo automático onde passa a se mover de forma autônoma. Modo Controle Manual: O carro inicia nesse modo. O carro passa para o modo manual onde aguarda uma instrução para executar.

Executa Comando Manual: O carro executa um comando manual, sendo esses comandos como: Frente (move o carro para frente), trás (move o carro para trás), esquerda (gira as rodas dianteiras para a esquerda), direita (gira as rodas dianteiras para a direita), parar (para os motores) e centro (retorna as rodas dianteiras para a posição inicial - centro). Os comandos para frente e para trás são configurados com o valor de PWM vinda do software enquanto os comandos de direita e esquerda possuem um ângulo fixo de deslocamento das rodas dianteiras.

Configura Valores: Configura o valor (ou um novo valor caso tenha sido chamado posteriormente) de PWM para o carro e configura quantas leituras a câmera irá fazer.

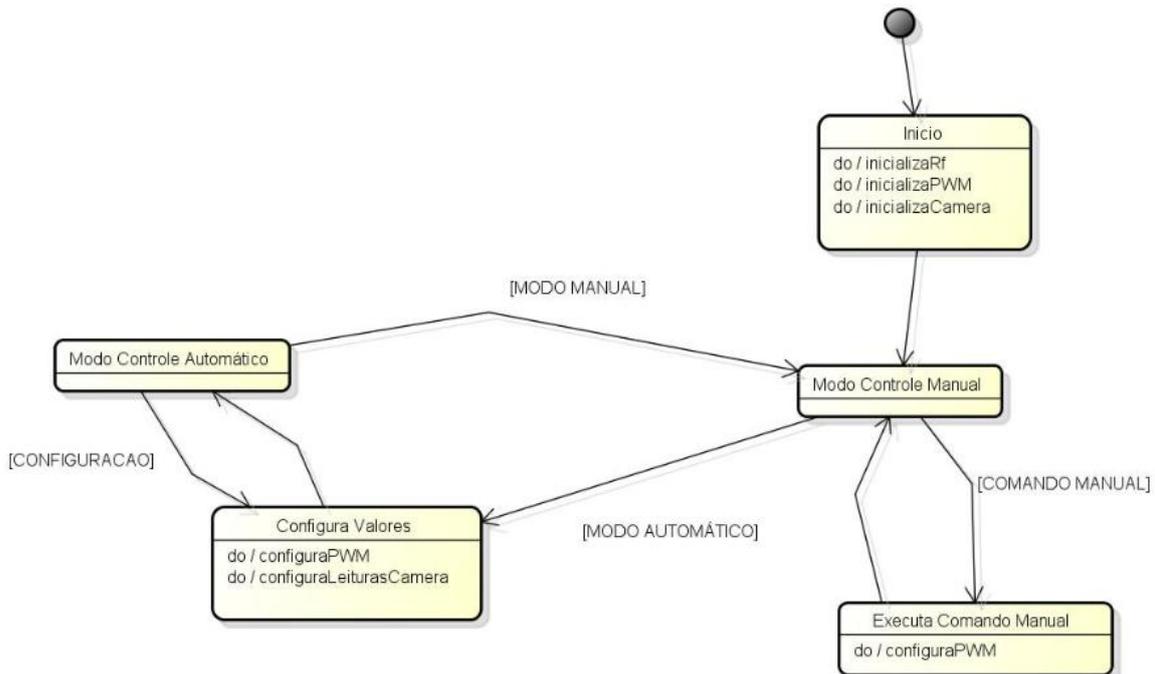



Figura 26: Diagrama de comunicação do Carro

A figura 27 representa os possíveis estados que a estação base pode assumir. Iniciar: Primeiro é iniciado o módulo RF que servirá para enviar e receber. Confere Instrução Software: Confere se chegou alguma instrução via serial. Confere Fila Mensagem: Confere se chegou alguma mensagem pelo RF.

Envia Comando: Se chegou alguma instrução pelo serial, envia essa instrução por RF para o carro.

Espera ACK: Após enviar um pacote, espera para receber um ACK. Caso não receba um ACK, reenvia o comando via RF até um limite de tentativas.

Erro: Se não chegar nenhum ACK dentro do número máximo de retransmissões, retorna uma mensagem de erro e volta a conferir se chegou alguma instrução via Serial. Recebe Dados Sensores: caso tenha chego algum pacote via RF, recebe esse pacote e envia por serial para o software.

A figura 28 representa o diagrama de estados do Arduino do Sistema Embarcado. Inicio: Inicializa o módulo RF e todos os sensores acoplados no arduino. Espera Sincronização: Espera receber um pacote de sincronização. Envia Dados Dos Sensores: Após receber um pacote de sincronização, passa a enviar os dados de todos os sensores a cada 50ms.

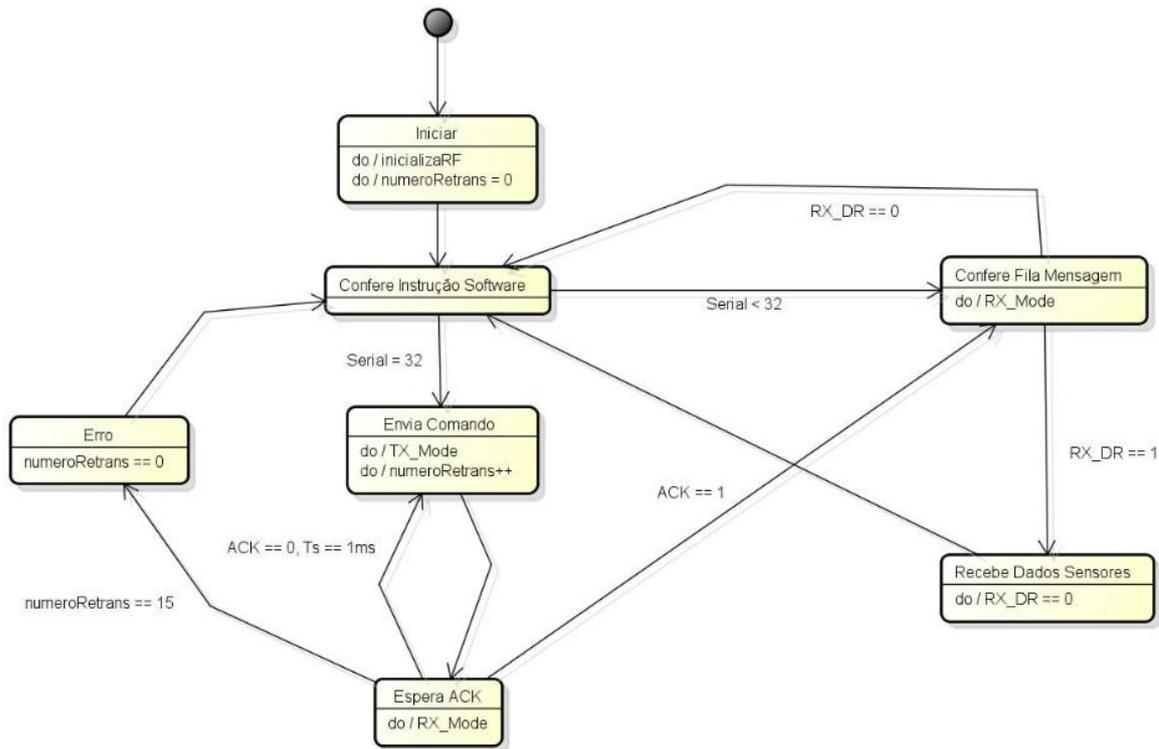



Figura 27: Diagrama de comunicação da estação base

Espera ACK: Após enviar o pacote, espera o recebimento de um ACK. Retransmite Dados Sensores: Caso não tenha recebido nenhum ACK, retransmite o pacote via RF. Espera 50ms: Espera durante 50 milisegundos antes de enviar outro pacote via RF. Como acoplar o arduino no carro foi uma solução repentina, seu protocolo de comunicação não foi completado, enviando somente a cada 50ms o valor dos sensores. Caso ocorra algum problema durante o envio, o arduino deve ser resetado.

6.4 Mudança no Sistema Embarcado

A equipe teve sérios problemas no desenvolvimento da comunicação. O maior problema foi em relação à comunicação do Sistema Embarcado (o carro) com a Estação Base. Quando o computador envia um pacote para o carro, este recebe e interpreta a mensagem e interpreta corretamente, mas o carro não consegue enviar uma mensagem de volta para a Estação Base.

O grupo acredita que seja um problema na placa FRDM-KL25Z ou na implementação do código SPI que comunica a placa com o módulo nRF24L01.

Como a comunicação é essencial para a estrutura do projeto e o tempo acabou ficando escasso, a equipe decidiu contornar o problema (seguindo sugestões dos professores da disciplina) incluindo um segundo microcontrolador no carro para interagir com o módulo RF. Como o módulo funcionou com um kit arduino (já utilizado na Estação-Base), o grupo decidiu utilizar um Arduino Uno (que os integrantes já possuíam) para a comunicação.

O arduino iria se comunicar com o módulo RF via SPI (protocolo utilizado pelo módulo nRF24L01)e enviar as informações via Serial para o KL25Z. Quando o carro recebesse uma notificação para enviar um dado para a estação base, o carro enviaria para o arduino via Serial e este enviaria para o computador via RF. Porém, o grupo teve dificuldades para

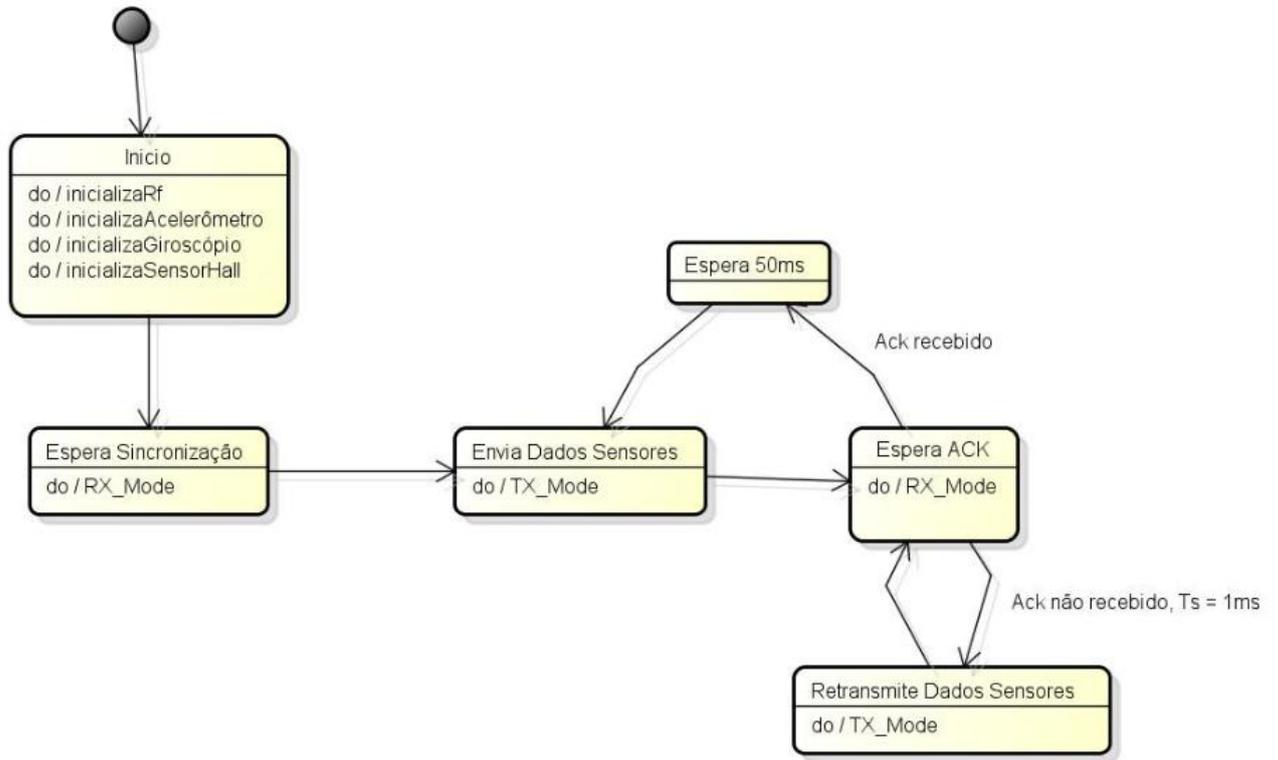



Figura 28: Diagrama de e estados do Arduino

enviar via Serial para o arduino, sendo que a comunicação Serial de ida (arduino para KL25Z) funcionou perfeitamente. A comunicação Serial de volta (carro para arduino) não funcionou e a equipe chegou a outro impasse.

Para continuar o desenvolvimento do projeto, pois a equipe perdeu mais tempo com esse problema, os professores sugeriram passar os sensores (MPU 6050 e o Sensor de Efeito Hall) para o arduino e este serviria somente para enviar o valor dos sensores para a Estação-Base.

Assim, o Arduino Uno foi acoplado ao carro com outro módulo RF, totalizando três módulos de comunicação (um módulo da Estação-Base, um módulo na placa KL25Z para receber os comandos do carro e um módulo no Arduino Uno para enviar os dados dos sensores para o computador). O diagrama de blocos em alto nível do projeto foi alterado ao adicionar o arduino. Podemos visualizar a nova estrutura do sistema de comunicação na figura 29.

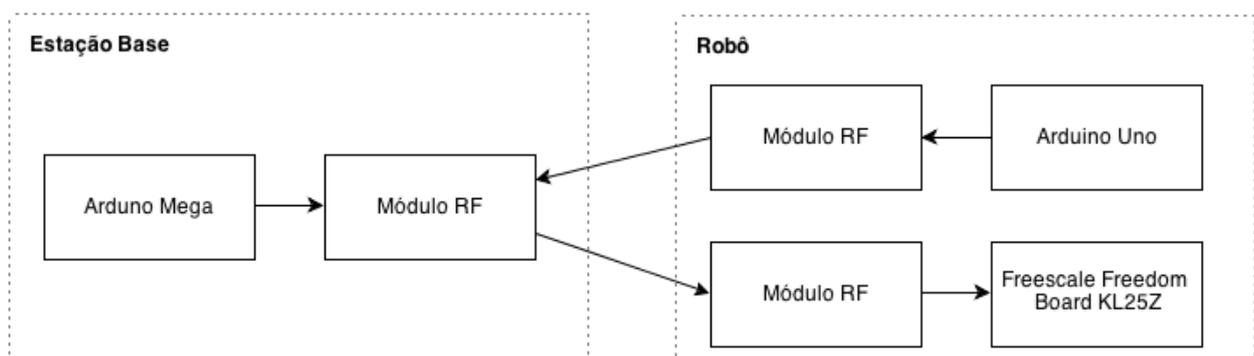



Figura 29: Diagrama simplificado do sistema de comunicação do projeto.

7 — Sistema embarcado

Na figura 30 temos uma foto do carro montado, com todos os sensores e componentes,que serão explicados a seguir, acoplados.

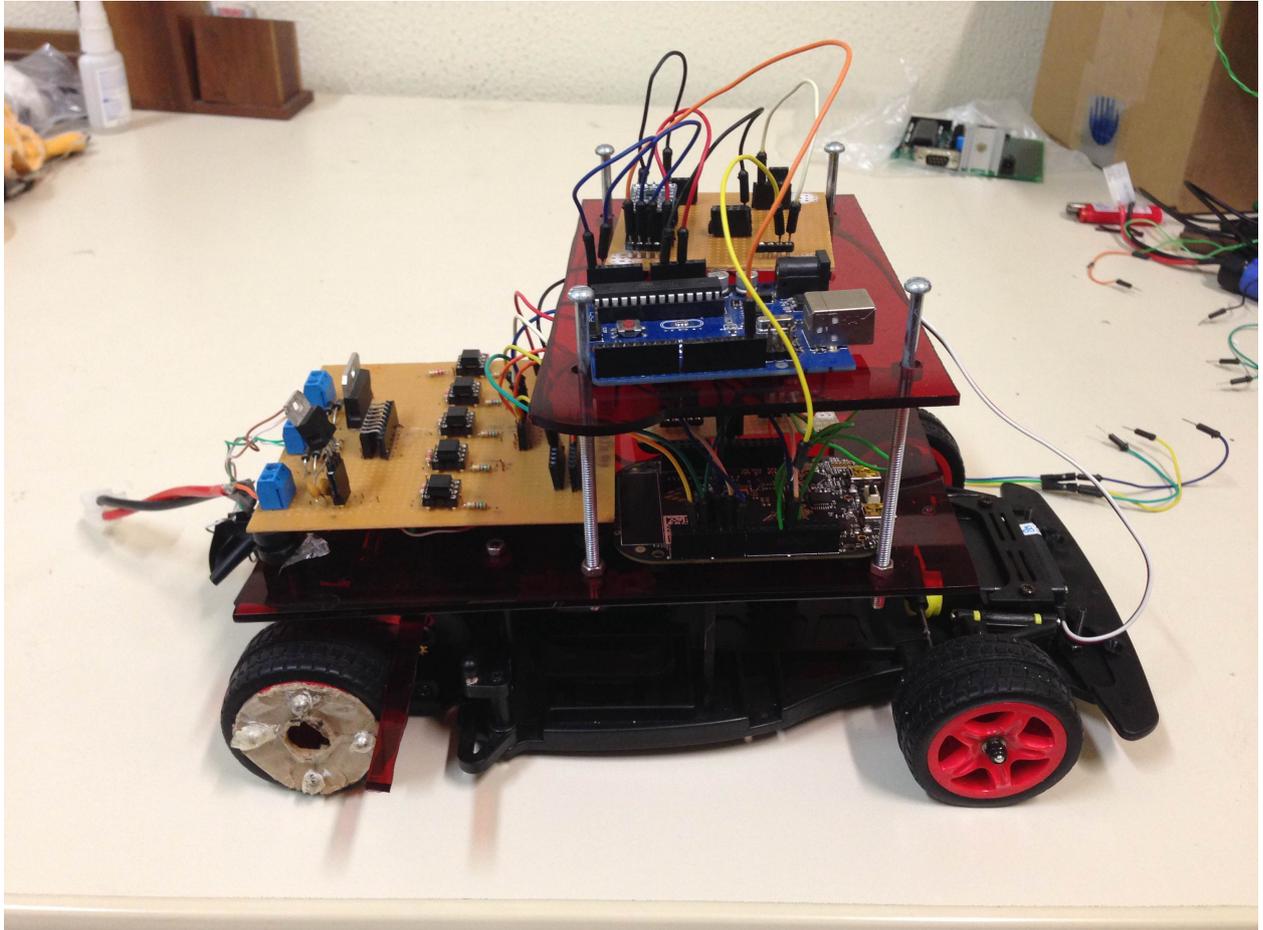



Figura 30: Carro montado

7.1 MPU-6050

7.1.1 Módulo MPU 6050

O módulo MPU 6050 contém um acelerômetro digital de 3 eixos e um giroscópio de 3 eixos além de conter um sensor de temperatura integrado ao dispositivo. O sensor foi produzido para possuir baixo consumo, baixo custo e alto desempenho principalmente para tablets e smartphones [6]. O sensor possui interface I^2C para comunicação externa e interna. Na comunicação interna (entre o sensor e seus registradores), o sensor utiliza o protocolo I^2C com uma velocidade de transmissão de 400kHz [6]. Para conversão dos dados analógicos captados pelo acelerômetro e pelo giroscópio, o módulo possui um conversor ADC de 16 bits para digitalizar a saída de cada sensor.

Estrutura

O esquemático do módulo MPU 6050 com suas devidas funções podem ser visualizadas na figura 31. O esquema mostra o top view do módulo que possuí 24 pinos. Os pinos NC não são conectados e os pinos RESV são reservados do módulo.

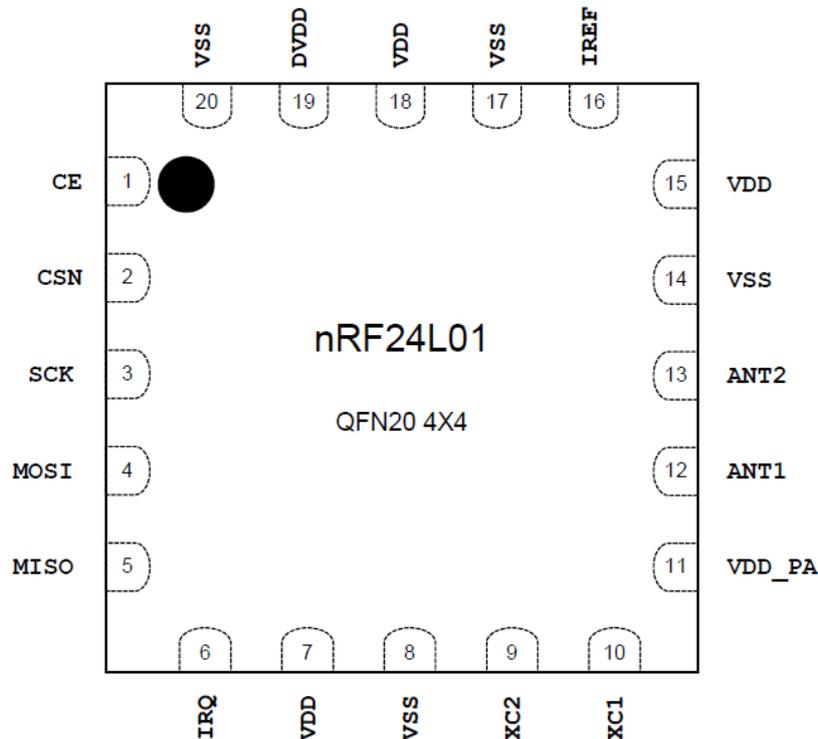



Figura 31: Configuração dos pinos do MPU 6050 [6]

Pino 1: CLKIN. Opção externa de referência para o clock, conectado no terra caso não seja utilizado. Pino 6: AUX_DA. Conexão I^{2}C master SDA auxiliar utilizado para sensores externos. Pino 7: AUX_CL. Conexão I^{2}C master SCL auxiliar utilizado para sensores externos. Pino 8: VLOGIC. Tensão de alimentação digital do circuito. Pino 9: AD0. Endereço LSB do slave. Pino 10: REGOUT. Filtro regulador do capacitor. Pino 11: FSYNC. Sincronização de frame digital, ligado ao terra caso não usado. Pino 12: INT. Pino de interrupção do sensor. Pino 13: VDD. Tensão de alimentação digital do sensor. Pino 18: GND. Terra do circuito. Pino 20: CPOUT. Ligação do capacitor do sensor. Pino 22: CLKOUT. Saída de clock do sistema. Pino 23: SCL. Pino que gera o clock para a conexão I^{2}C. Pino 24: SDA. Pino que transmite os dados para a conexão I^{2}C. O módulo que a equipe utilizará no projeto já vem com o sensor soldado na placa. Por isso a maioria dos pinos apresentados acima não foram utilizados no projeto (como os pinos AUX_DA ou AUX_CL) e não serão detalhados. A placa possui oito pinos, com espaçamento padrão para facilitar a conexão externa (com uma protoboard ou placa furada) e utiliza somente os pinos mais importantes do sensor enquanto os outros são ligados diretamente no terra. A figura 32 ilustra os pinos do MPU 6050 já soldado na placa.

Para a ligação do módulo com o microcontrolador FRDM-KL25Z foram utilizados os pinos PTE0 e PTE1 para conectar os pinos de SDA e SCL do módulo respectivamente. O pinos 5V USB OUT foi usado para alimentar o módulo enquanto os pinos de GND foi conectado ao GND da placa.

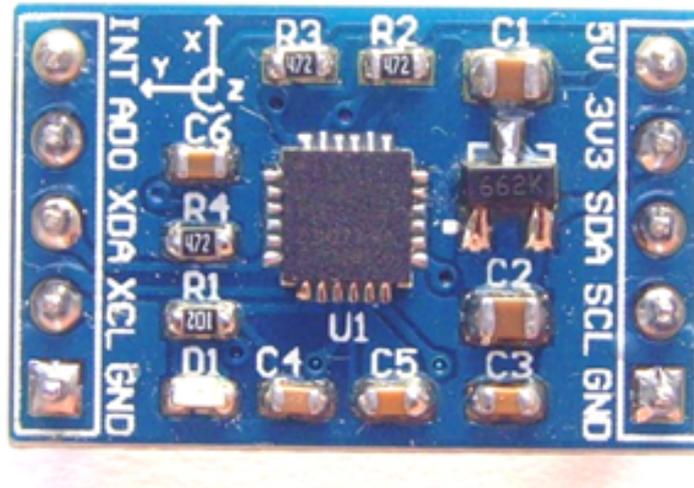



Figura 32: Módulo MPU 6050.

7.1.2 Funcionamento do Módulo MPU 6050

Para a implementação do software, alguns registradores foram utilizados para configuração dos sensores enquanto outros foram lidos diretamente. O funcionamento do módulo segue o seguinte modelo:

1) Cada eixo dos sensores (eixos X, Y e Z do acelerômetro e giroscópio) lê seu respectivo valor.

2) Esse valor analógico passa por um conversor ADC para converter em digital.

3) Após a conversão, o eixo espera um sinal para enviar esse valor para um registrador apropriado.

4) O valor pode ser armazenado ou em um registrador denominado Sensor Register, ou em uma fila denominada FIFO (First In First Out).

5) O módulo envia esses dados via I^2C para o Master pela saída SDA utilizando o clock SCL para sincronizar os dispositivos.

O diagrama de blocos pode ser visualizado na figura 33, onde mostra o caminho superficial dos dados no circuito.

O módulo possui 3 eixos do giroscópio ligados a um conversor ADC. Quando o giroscópio rotaciona em um desses eixos uma vibração pode ser captada pelo sensor. Esse sinal resultante é amplificado, demodulado e filtrado para produzir uma tensão proporcional à variação de ângulo ocorrida. O sinal passa por um conversor ADC para cada eixo. A escala do giroscópio pode ser configurada em 250, 500, 1000, 2000 graus por segundo [6].

Já o acelerômetro possui 3 eixos também acoplados a um conversor ADC. Utilizando um sensor capacitivo interno, o acelerômetro consegue captar a variação de gravidade. Quando o sensor está parado em uma posição plana, irá medir 0g para o eixo X, 0g para o eixo Y e 1g para o eixo Z. A escala do acelerômetro pode ser configurada em 2g, 4g, 8g ou 16g [6].

O módulo possui registradores para auto teste além de registradores específicos para interrupção. Alguns pinos ou interfaces não foram utilizadas no projeto e portanto não serão detalhadas nesse documento.

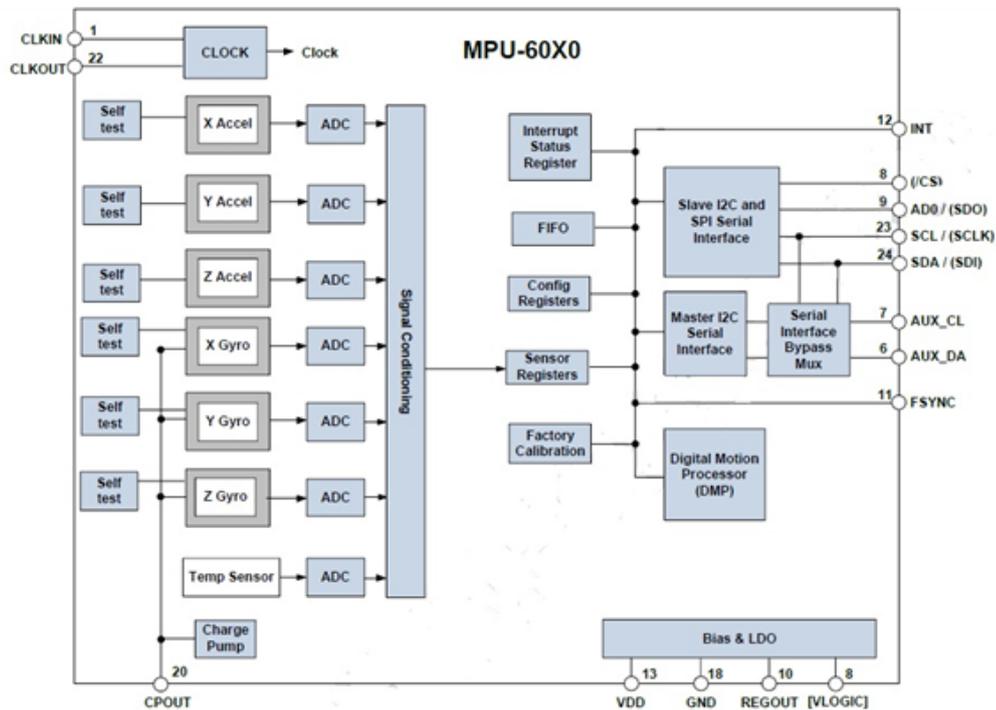



Figura 33: Diagrama de blocos [6].

Escrita de Bytes

Na escrita, primeiramente o Master envia um bit para inicializar a comunicação (bit de start condition), seguido do endereço 7 bits do módulo com quem se deseja comunicar (0x68). O oitavo bit desse endereço deve ser um bit de escrita (bit 0), representando que o Master deseja escrever no módulo. Após o retorno do ACK do dispositivo Slave, o Mestre envia o endereço do registrador a qual deseja escrever e passa a enviar o valor que se deseja escrever naquele registrador. A figura 34 ilustra o protocolo de escrita da interface I^2C no módulo.

Master	S	AD+W		RA		DATA		P
Slave			ACK		ACK		ACK	

Figura 34: Protocolo de escrita de bytes no módulo MPU 6050.

Leitura de Bytes

Para leitura, o mestre deve inicialmente seguir os mesmos procedimentos da escrita. O mestre envia um bit de start condition seguido do endereço 7 bits (ox68) do módulo, sendo o oitavo bit representado por um bit de escrita (0), informando que o mestre deseja escrever no módulo. Após o ACK, o Master envia o endereço do registrador a qual se deseja ler (e escreve esse endereço no módulo, por isso a primeira conexão com o módulo possui um bit de escrita). O mestre inicia uma nova conexão com o módulo e envia novamente o endereço 7 bits do módulo, mas dessa vez o oitavo bit é representado por um bit de leitura (1), indicando que o mestre deseja ler. Após o ACK, o módulo passa a receber o valor, lendo o registrado desejado. A figura 35 ilustra o protocolo de leitura do modulo MPU 6050.

Master	S	AD+W		RA		S	AD+R			NACK	P
Slave			ACK		ACK			ACK	DATA		

Figura 35: Protocolo de leitura de bytes no módulo MPU 6050.

Inicialização

Para inicializar o módulo é preciso escrever em alguns registradores de configuração do sensor. Primeiramente, é preciso configurar o giroscópio e o acelerômetro para as escalas ideais utilizadas no projeto. O giroscópio irá funcionar com uma escala de 250 graus por segundo, suficiente para os requisitos do projeto.

O acelerômetro também não irá precisar de uma escala muito grande sendo que 2g já atenderá as especificações do projeto. A escala pequena se deve ao fato de que o carro irá desprezar variações muito pequenas, considerando variações um pouco maiores.

O giroscópio pode ser configurado pelo registrador 1B (ou 27 em decimal) onde configurando os bits 3 e 4 do registrador como zero podem selecionar a escala de 250 graus por segundo. Semelhante ao giroscópio, o acelerômetro pode ser configurado pelo registrador 1C (28) onde os bits 3 e 4 são configurados como zero para selecionar a escala de duas gravidades.

O módulo ainda precisa ser configurado para utilizar um clock externo ou interno. Como padrão, o sensor vem configurado para utilizar um clock interno de 8MHZ, que utiliza menos consumo.

O datasheet do módulo MPU 6050 aconselha utilizar o clock interno referente a um eixo do giroscópio (clock que referencia ao eixo X, Y ou Z) ou um clock externo ao sistema, para aumentar a estabilidade. Para isso, é preciso configurar os três primeiros bits do registrador Power Management 1 (registrador 6B), como 1, utilizando o eixo X do giroscópio como referência. Esse mesmo registrador é utilizado para retirar o módulo da configuração de sleep (durmindo) selecionando o bit 6 do registrador como zero.

A ultima configuração feita no sensor foi a sincronização da saída do acelerômetro e do giroscópio. O acelerômetro possui uma velocidade de saída de 1kHz enquanto o giroscópio pode variar dependendo da configuração utilizada. A saída do giroscópio permanece em 1kHz quando o DLPF (Digital Low Pass Filter) não está configurado em zero, ou seja, quando ele está ativo.

Entretanto quando o DLPF está ativo, ocorre um delay (atraso) na saída dos dados por conta da filtragem ocorrida. A figura 36 ilustra a variação ocorrida com o DLPF. Ao desabilitar o DLPF, com o objetivo de contornar esse delay, a saída dos dados do giroscópio passa para 8kHz, oito vezes maior que o acelerômetro. Para que isso não ocorra, o módulo fornece um registrador denominado Sample Rate Divider (19 em hexa) que permite dividir a saída do giroscópio. A equação utilizada para divisão é a seguinte:

Saída Giroscópio = Saída Padrão do Giroscópio / (1 + Sample Rate Divider)

Onde a saída Padrão do Giroscópio é igual a 8 kHz e a Sample Rate Divider é o numero desejado para a divisão. Logo, para configurar o giroscópio como 1 kHz, o Sample Rate Divider foi escrito com o valor sete, enquanto o DLPF é configurado como zero.

Leitura dos Valores

Para a leitura dos dados de cada eixo dos sensores, foram lidos os registradores de 3B a 40 (em hexadecimal) para o acelerômetro e de 43 a 48 para os eixos do giroscópio. Os

DLPF_CFG	Accelerometer (F_s = 1kHz)		Gyroscope		
	Bandwidth (Hz)	Delay (ms)	Bandwidth (Hz)	Delay (ms)	Fs (kHz)
0	260	0	256	0.98	8
1	184	2.0	188	1.9	1
2	94	3.0	98	2.8	1
3	44	4.9	42	4.8	1
4	21	8.5	20	8.3	1
5	10	13.8	10	13.4	1
6	5	19.0	5	18.6	1
7	RESERVED		RESERVED		8

Figura 36: Configuração do DLPF no módulo MPU 6050.

registradores foram lidos seguindo o protocolo I^2C descrito acima. Os valores captados por cada eixo possuem 16 bits de tamanho e como cada registrador possui no total 8 bits de tamanho, o módulo separa dois registradores para cada eixo, sendo um registrador para os bits mais significativos do valor (MSB) e um registrador para os bits menos significativos (LSB). Na figura 37 podemos observar os registradores de leitura do acelerômetro. A sensibilidade do acelerômetro para um escala de 2g é de 16384 LSB/g enquanto o giroscópio possui uma sensibilidade de 131 LSB/grau/segundo.

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
3B	59	ACCEL_XOUT[15:8]							
3C	60	ACCEL_XOUT[7:0]							
3D	61	ACCEL_YOUT[15:8]							
3E	62	ACCEL_YOUT[7:0]							
3F	63	ACCEL_ZOUT[15:8]							
40	64	ACCEL_ZOUT[7:0]							

Figura 37: Registrador de leitura do acelerômetro.

7.2 Câmera de Linha

7.2.1 Aspectos Gerais

Para detectar o trajeto definido pela faixa preta do cicuito foi utilizada a camera de linha TSL1401R LF da TAOS. Na Figura 38 temos a imagem da câmera.

A câmera é conectada com 5 pinos:

- GND
- V_{DD}
- CLK
- SI
- AO

A alimentação pode ser de 3 até 5 volts. No nosso projeto utilizarmos a porta que fornece 3,3V, pois o shield do kit de desenvolvimento tem entrada para a câmera que por sua vez se conecta a porta de 3,3V.

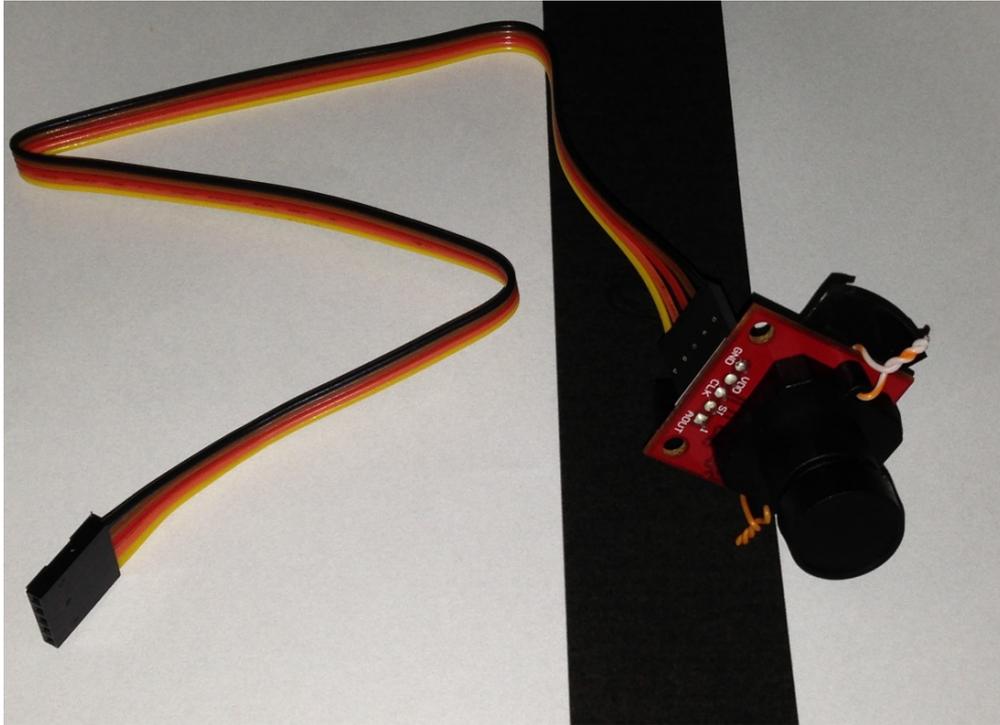



Figura 38: Câmera de Linha

Para fazer a leitura é necessario ter um clock. A cada pulso a saida pula para o proximo pixel a ser exibido. O clock maximo é de 8MHz, porém no site da Freescale é recomendado um clock entre 5 e 8 kHz. Escolhems utilizar um período de 168μs, que resulta em um clock de aproximadamente 5952 Hz.

O pino SI é usado para dar inicio a captura da imagem. Ao gerar um pulso a camera começa a enviar a saída pelo pino AO (Analog Output).

O pino AO (Analog Output) é usado para enviar a saída para o microcontrolador. Os pixels são passados de forma serial, a cada pulso do clock a tensão de saída varia, de acordo com o proximo pixel a ser enviado. Quanto menor a tensão mais escuro foi o resultado. Ao fim dos 128 pixels a saída volta para zero.

Para entermos melhor podemos analisar a Figura 39, retirada do datasheet que mosta as ondas em relação ao tempo.

Há algumas coisas importantes a serem observadas.

- Enquanto SI estiver em alta, é necessário que um pulso de clock vá para alto. Como usa um detector de borda de subida, SI será detectado como alto durante a mudança e assim se dara inicio a medição.
- É necessário que SI esteja em baixa antes da leitura começar.
- O tempo total de leitura são 129 ciclos do clock. Um para a detecção do SI e um para cada pixel.
- Após 18 ciclos de clock começa a integração e só termina quando Si é ativado novamente.

7.2.2 Integração

A integração é um conceito muito importante. Durante o tempo de integração a luz é amostrada e a energia é acumulada em cada capacitor (cada pixel tem um capacitor). A integração ocorre depois de 18 clocks e vai até um novo sinal de SI. Um número de 129 clocks

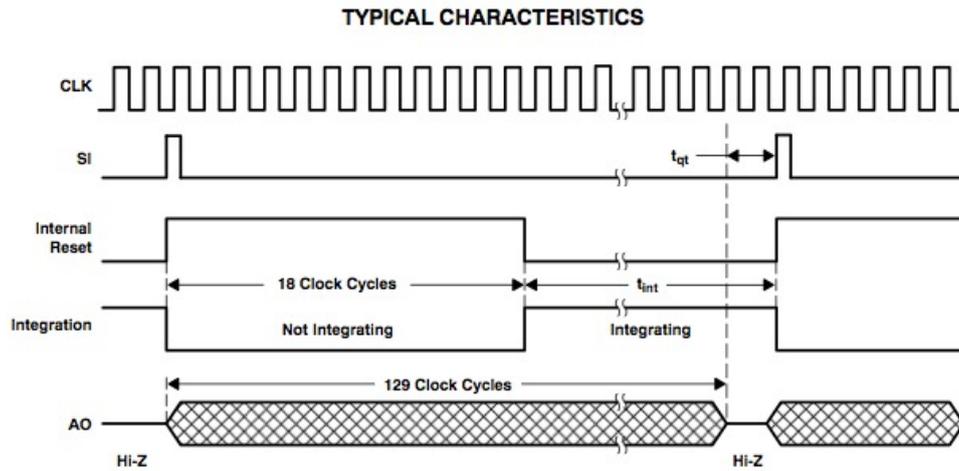
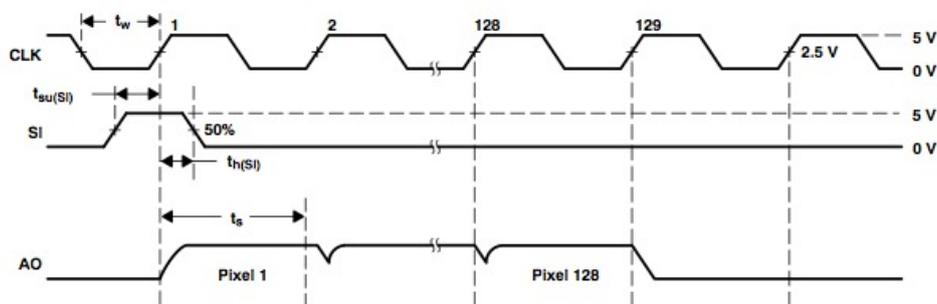



Figura 39: Ondas em relação ao tempo.

devem ocorrer antes de um novo ciclo de leituras. Mas não é necessário que logo após os 129 clocks ocorra um novo ciclo. Pode se esperar até 100 ms até uma nova leitura. Esse tempo é muito importante, pois quanto mais tempo deixado maior é a saída no proximo ciclo. Isso pode ser usado em ambientes com baixa luminosidade.

O tempo de integração pode ser obtido com a seguinte fórmula:

$$T = (1/freq) * (n - 18) + 20$$

Sendo n o número de pixels e freq a frequencia do clock.

Para o caso da nossa equipe temos como frequência 5952 HZ e 128 pixels. Substituindo na equação temos:

$$T = (1/5952) * (128 - 18) + 20$$

$$T = 18500\mu s = 18,5ms$$

7.2.3 Testes

Utilizando um osciloscópio e uma folha com uma faixa preta foram feitos testes com a câmera de linha. As Figuras 40,41 e 42 mostram o resultado.

O resultado dos testes foi muito bom, podemos ver claramente onde está localizada a faixa preta. Para complentar esse teste foi feito um teste diretamente com a placa KL25Z. Ao invés da saída ser colocada no osciloscopio medimos e para valores altos imprimimos o número um e para valores baixos imprimimos o número 0 (nos testes utilizamos o separador

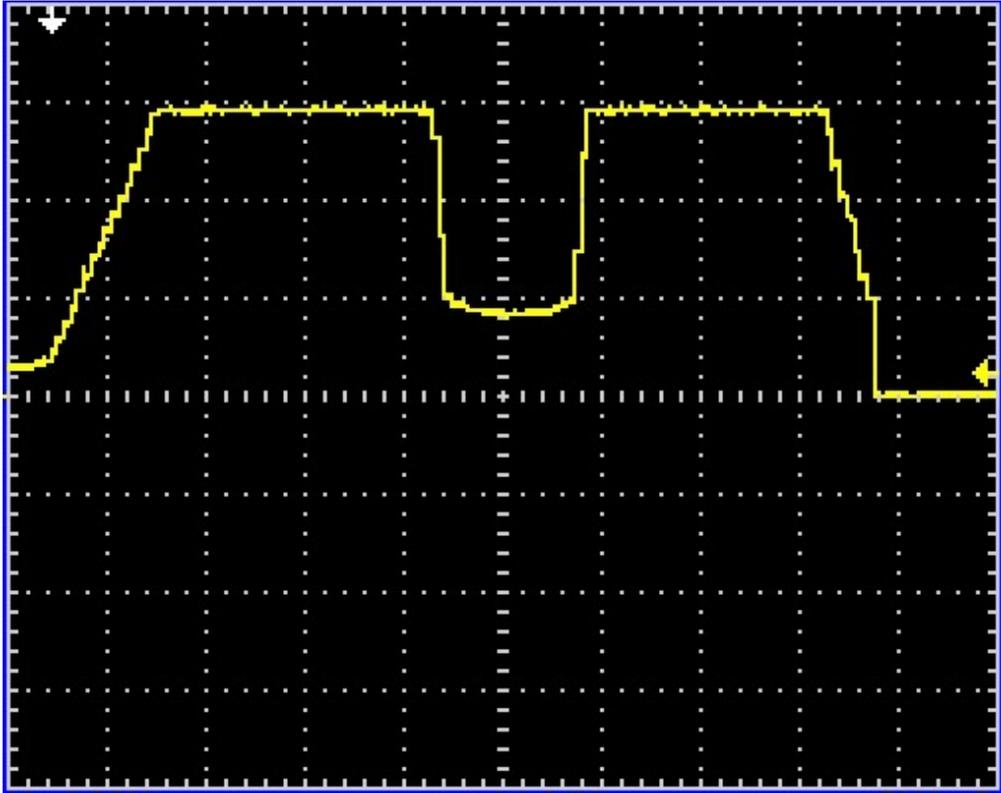



Figura 40: Saída camera com a faixa preta no centro

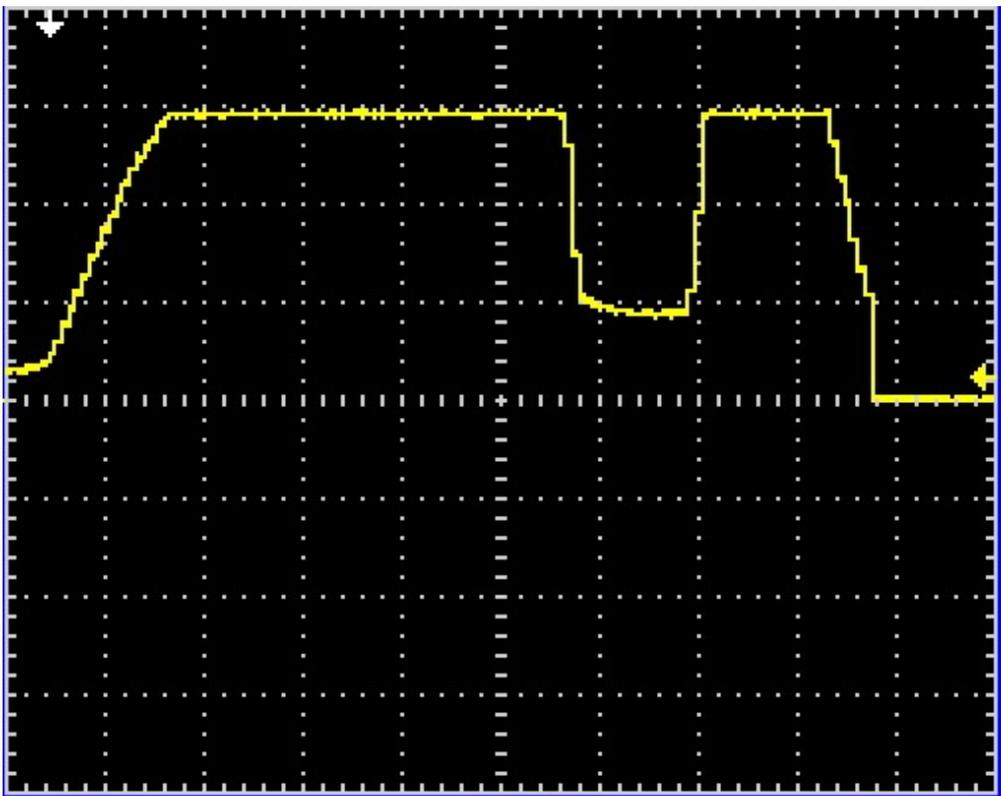



Figura 41: Saída camera com a faixa preta a direita

0.9 para claro e escuro). Para três leituras consecutivas podemos ver o resultado na figura 43.

Figura 42: Saída camera com a faixa preta a esquerda

Figura 43: Saída camera imprimindo 1 para claro e 0 para escuro

Nesse teste a leitura também se mostrou eficiente. Outro item que podemos notar é que em ambos os testes os pixels dos cantos não condizem com a realidade. Após verificar essa falha, verificamos na documentação da câmera e constatamos que de fato os pixels das extremidades devem ser ignorados.[8]

7.3 Módulo nRF24L01

7.3.1 Protocolo SPI

O SPI (Serial Peripheral Interface) foi originalmente desenvolvido pela Motorolanos e adotado por vários fabricantes ao longo dos anos. Atualmente, o barramento SPI é usado principalmente nos microcontroladores e seus dispositivos periféricos imediatos. É comumente encontrado em telefones celulares, PDAs e outros dispositivos móveis que comunicam dados entre a CPU, teclado, tela e chips de memória (AGUIAR et al, s.d.). O SPI consiste em três barramentos principais, envolvendo um dispositivo Mestre (ou Master) e um ou mais dispositivos Escravos (ou Slave). O SCLK (Serial Clock) é gerado pelo Master e serve para sincronizar a transferência de dados. O Sinal MOSI (Master Out Slave In) transporta os dados do Master para Slave, enquanto o Sinal MISO (Master In Slave Out) transporta os dados do Slave para Master. Cada Slave é selecionado por um nível lógico baixo ('0') através da linha CS (Chip Select). Os dados sobre este barramento pode ser transmitidos a uma razão de quase zero bits por segundo até 1Mbits/s. Os dados são transferidos em blocos

de 8bits, no qual o bit MSB(Most Significant Bit) é transmitido primeiro (AGUIAR, et al, s.d.). A transmissão dos dados ocorre da seguinte maneira. O Mestre primeiro envia um sinal SCLK para sincronizar os dispositivos. O Mestre envia um bit na linha MOSI e o Slave lê a mesma linha. O Slave envia um bit pela linha MISO e o Mestre lê essa linha. Quando não ha mais dados para serem transmitido, o Master para alternando seu Clock.

7.3.2 Módulo nRF24L01

O módulo nRF24L01 é um sensor transceiver (envia e recebe dados) que funciona por rádio frequência. Ele foi produzido para operar em uma faixa de frequência de 2,4 GHz seguindo as normas da ISM (Industrial, Scientific and Medical radio bands). O módulo utiliza o protocolo SPI para comunicação, utilizando vários registradores internos para configuração e leitura dos dados (NORDIC, 2008). Ele utiliza um protocolo de envio e recebimento de dados denominado ShockBurst, onde realiza o gerenciamento dos pacotes e respostas, além de possuir uma fila para controle dos dados. O módulo pode enviar de 1 até 2 Mbps (mega bits por segundo)e possui modos de baixa corrente e baixo consumo. O módulo funcionado com uma tensão de entrada de 1,9 até 3,6 Volts, possui pinos de RX e TX (sendo módulo duplex), possui um cristal de 16 MHz, pode enviar até 32 bytes por pacote e pode funcionar como modo multiceiver (recebe dados de vários módulos) [7].

Estrutura

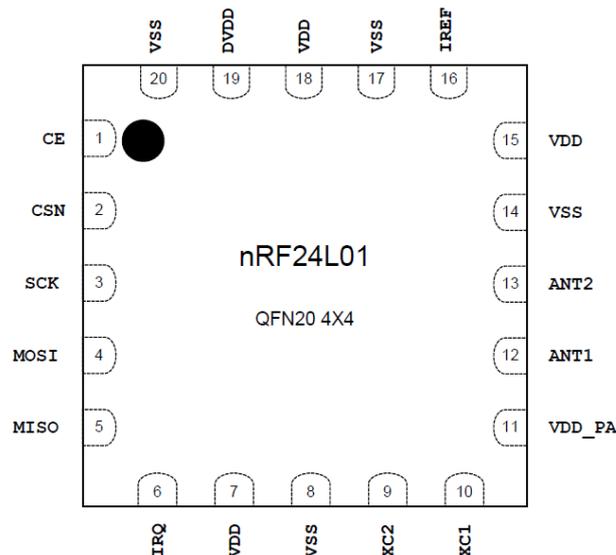
A estrutura do módulo funciona baseado no protocolo SPI e as funções de cada pino do chip podem ser visualizadas na figura 44, onde mostra cada pino separadamente. Os pinos descritos como VSS são a referência para o terra do módulo e os pinos descritos como VDD são as tensões de alimentação do módulo, que podem variar de 1,9V até 3,6V em nível DC.

Fonte: Adaptado de NORDIC, 2008

Figura 44: Estrutura do módulo nRF24L01.

Pino 1: CE. Pino que seleciona se o módulo estará no estado TX (enviando) ou RX (recebendo). Pino 2, 3, 4 e 5: CSN, SCK, MOSI e MISO. Pinos para utilizar o protocolo de comunicação SPI, onde CSN é o chip select, SCK é o clock, MOSI e MISO são os barramentos de dados. Pino 6: IRQ. Pino de interrupção do módulo. Pino 9 e 10: XC2 e XC1. Pinos de entrada do cristal oscilador de 16 MHz. Pino 11: VDD_PA. Tensão (1,8V) interna do módulo direcionada para um amplificador, devendo ser ligada aos pinos ANT1 e ANT2. Pino 12 e 13: ANT1 e ANT2. Antena 1 e Antena 2 do módulo. Pino 16: IREF. Referência para corrente, conecta um resistor de 22k para no terra. A placa com o chip soldado possui 8 pinos disponíveis, sendo esses pinos já descritos acima e pode ser visualizada na figura 45. Os pinos CSN, SCK, MOSI e MISO são os pinos referentes a comunicação SPI do módulo, IRQ é o pino de interrupção, CE o chip select e VCC e GND são os pinos de alimentação e terra respectivamente.

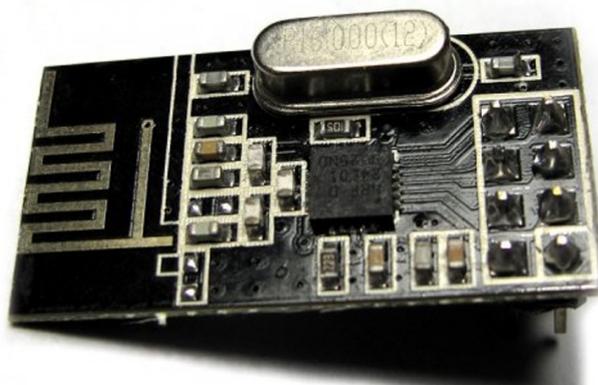



Figura 45: Módulo nRF24L01.

Funcionamento

O módulo possui quatro estados básicos de operação, o modo Power Down, Standby, modo TX e modo RX. O modo Power Down é inicializado quando uma tensão igual ou maior que 1,9 Volts alimenta o circuito. Nesse modo, o módulo estará desativado consumindo o mínimo de corrente, mas mantendo a comunicação com os registradores via SPI ativada. O Power Down é ativado quando um bit chamado PWR_UP é configurado como zero. O modo Standby também consume pouca corrente, mas mantém parte do cristal oscilador ativo. Após o módulo enviar ou receber algum dado, ele retorna para o modo Standby, esperando outra ativação. O modo standby é ativado quando o bit PWR_UP é configurado como 1 e o bit CE é configurado como zero. O modo RX é ativo quando o módulo é configurado para receber um dado. Nesse modo, o transceiver demodula o sinal do canal RF procurando por um pacote que seja válido. Se o pacote é válido (passando por um protocolo de identificação do pacote), o módulo coloca esse pacote em uma fila denominada FIFO e lê os dados desse pacote. Caso a fila esteja cheia, o pacote é descartado. O modo RX é ativado quando o bit PRIM_RX é configurado como 1 e CE está ativado. O modo TX é ativo quando o transceiver é configurado para enviar um dado. O módulo permanece nesse estado até terminar de enviar o pacote. O módulo pode voltar para o estado standby e espera outra ativação ou passar para o estado standby 2, descrito a seguir. O modo TX é ativado quando o bit PRIM_RX é configurado como zero, CE está ativado. Existe um quinto estado do módulo denominado standby 2. O módulo passa para este estado quando a fila de envios TX FIFO está vazia e CE está ativado. Isso representa que o módulo está esperando um pacote ser formado para enviar. Logo, quando um pacote está pronto para envio, o módulo passar do estado standby

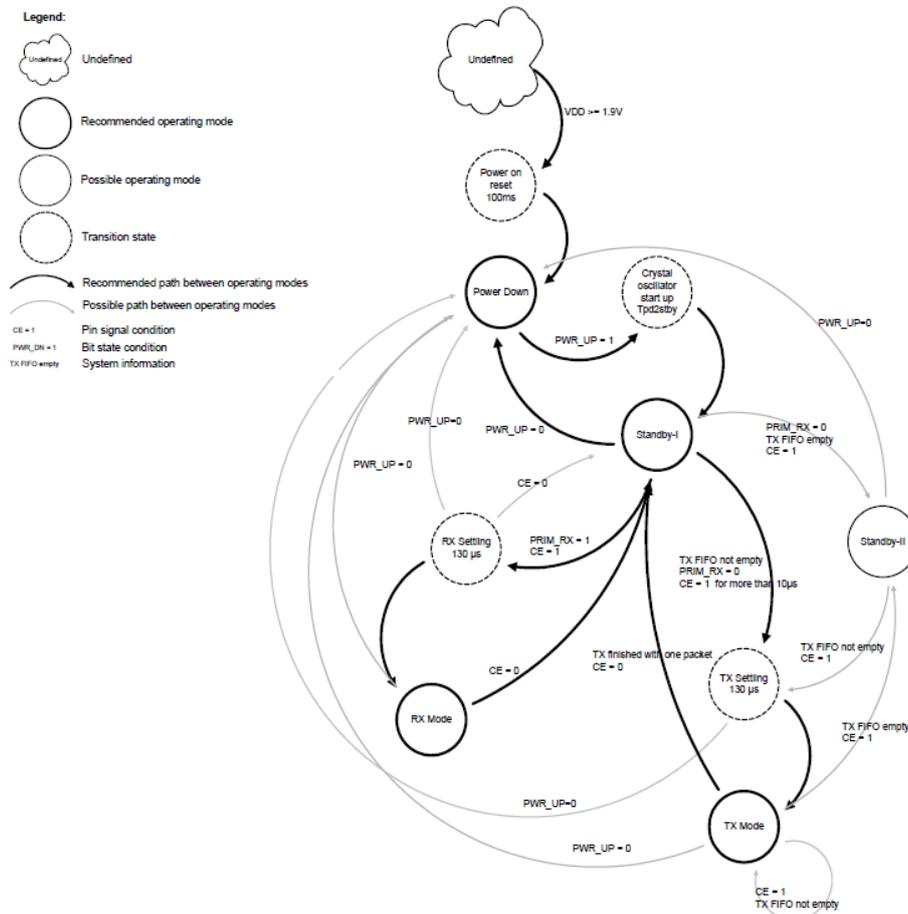
2 para o modo TX e caso o bit CE continue ativado e fila volte a ficar vazia, o módulo retorna para o estado de standby 2. A figura 46 demonstra todos os estados do módulo e o tempo de espera para a transição de cada estado.

Fonte: Adaptado de [7]

Figura 46: Transição de estados do módulo.

7.3.3 Protocolo Enhanced ShockBurst

É um protocolo de comunicação entre dois módulos nRF24L01. O protocolo possui um sistema de retransmissão de dados (com timing automático), envio de ACK e recebimento de pacotes. Ele permite o módulo permanecer em baixo consumo e maior desempenho [7]. Quando ativo, o protocolo segue a mesma sequência dos modos de operação descritos no tópico anterior, mas com uma pequena mudança. Abaixo segue o passo a passo do protocolo. Modo TX: 1) Quando o módulo está no estado TX (envio de dados) ele envia o pacote e passa para o modo RX esperando o recebimento de um ACK. 2) Ao receber o ACK, o módulo passa para o estado TX e continua o envio de pacotes caso a fila de envio (TX FIFO) não esteja vazia e o bit CE permaneça ativo (descrito acima). 3) Caso a fila fique vazia e o bit CE esteja ativo, o módulo passa para o estado standby 2, onde espera um pacote entrar na fila para enviar. 4) Caso não receba um ACK durante determinado período de tempo (podendo ser configurado nos registradores) o módulo reenvia o pacote novamente e espera o recebimento de um ACK. 5) Se o número de reenvio igualar um valor determinado na configuração do módulo, ocorre uma interrupção que deve ser tratada posteriormente. 6)

Se o bit CE passar para zero, o módulo volta para o estado de stanby e espera instruções. Modo RX: 1) No modo RX, o módulo espera pelo recebimento de pacotes. 2) Ao receber um pacote, o módulo analisa de esse pacote é válido e coloca esse pacote em uma fila de leitura, RX FIFO. 3) O módulo passa para o modo TX e envia um ACK. 4) Caso o bit CE seja desativado, o módulo passar para o estado standby e espera outras instruções.

Formato do Pacote

O formato do pacote enviado pode ser visualizado na figura 47.

Fonte: Adaptado de [7]

Figura 47: Formato de pacote no protocole Enhanced ShockBurst.

O preamble serve para detectar os níveis de 1's e 0's no receptor, assim tendo como obejtivo estabilização e servindo para garantir que existam transições suficiente. Ele pode ser configurado com 1 byte possuindo dois valores somente (10101010 ou 01010101). O Address é o endereço do módulo e serve para garantir a chegada do pacote no módulo apropriado. Pode ser configurado como 3, 4 ou 5 bytes. Logo o endereço é uma continuação do Preamble e completa o mesmo. Quando um pacote é enviado, é enviado com o primeiro valor do preamble e o endereço. O próximo pacote será enviado com o segundo valor de preamble e o mesmo endereço, e assim sucessivamente. O Packet Control Field possui 9 bits onde os seis primeiros bits configuram o tamanho do dado (ou payload) que será enviado. Esse tamanho pode variar entre 0 bytes até 32 bytes. Os próximos dois bits referenciam o ID do pacote ou identificação do pacote. Eles servem para detectar se o pacote recebido é novo ou retransmitido, assim prevenindo que ocorram dados repetidos. O ultimo bit pode configurar se o pacote enviado deseja receber ou não um ACK como resposta. A figura 48 ilustra o modelo do Packet Control Field.

Fonte: Adaptado de [7]

Figura 48: Formato do Packet Control Field no pacote.

O Payload contém os dados enviados por cada pacote e pode variar entre 0 bytes até 32 bytes. Por ultimo, o CRC é um mecanismo de detecção de erro utilizado pelo protocolo para informar se pacote enviado não se alterou durante o processo de envio.

7.4 Driver dos Motores

O controle dos motores DC das rodas traseiras do carro iria ser realizado pelo shield da Freescale, que contém dois chips de driver de motor (MC33887APVW). No entanto, um dos chips estava apresentando um comportamento anômalo, impedindo o acionamento correto de um dos motores.

Abandonamos, então, a utilização do shield e decidimos projetar uma PCB para controlar os motores.

7.4.1 Projeto do Driver

Para impedir que a grande quantidade de ruído gerada pelos motores DC afete o microcontrolador, resolvemos isolar o circuito de potência do circuito do microcontrolador. Este isolamento é realizado de duas formas: alimentação por fontes DC distintas e isolamento dos pontos de comunicação entre os dois circuitos por meio de optoacopladores.

Para controlarmos o sentido e a direção dos motores, necessitamos de duas pontes-H. Ao invés de utilizarmos dois chips 33887AP como feito no shield, decidimos por uma opção mais compacta, um único chip L298N, que contém duas ponte-H integradas capazes de suprir a potência necessária para o funcionamento em capacidade máxima dos motores.

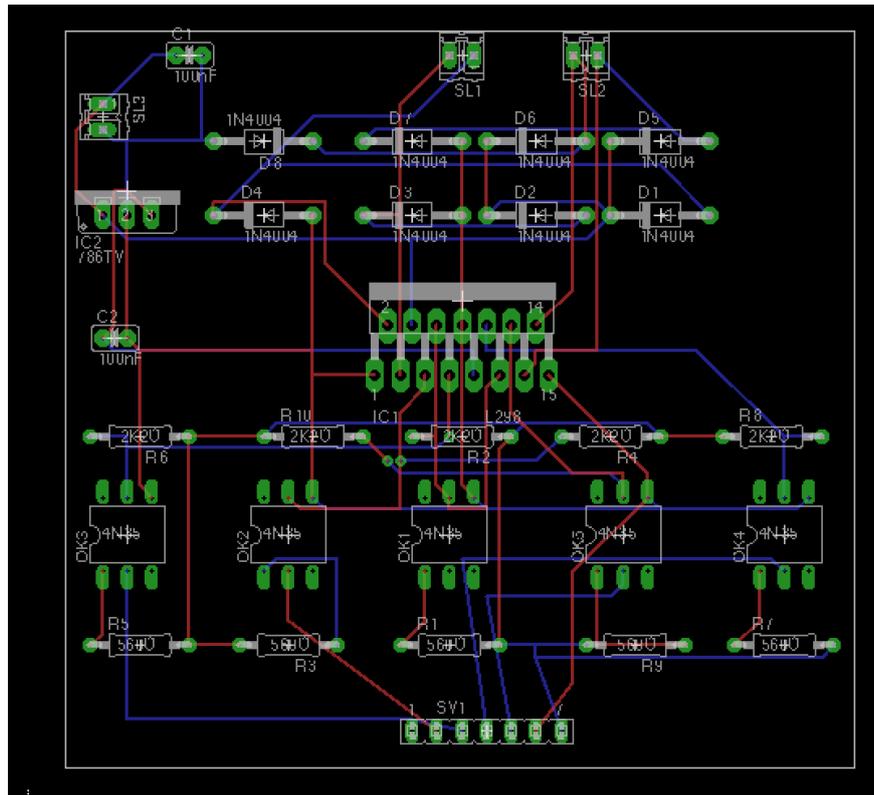
Como os motores DC são cargas indutivas, eles armazenam energia e resistem à variação de corrente. Caso haja uma variação brusca de corrente, como no caso de um desligamento da ponte-H previamente ativa, os motores irão gerar um pico de tensão que pode danificar o driver causada pela corrente de *flyback*. Para evitar danos, utilizamos diodos de proteção (1N4007) entre os motores e a fonte de potência, que ficam ativos em casos de sobretensão, fazendo com que a fonte absorva todo ruído gerado e proteja o driver dos motores. O esquemático do circuito projetado pode ser visualizado na figura 50 e o projeto em circuito impresso do circuito pode ser visualizado na figura 49.

Fonte: Autoria própria

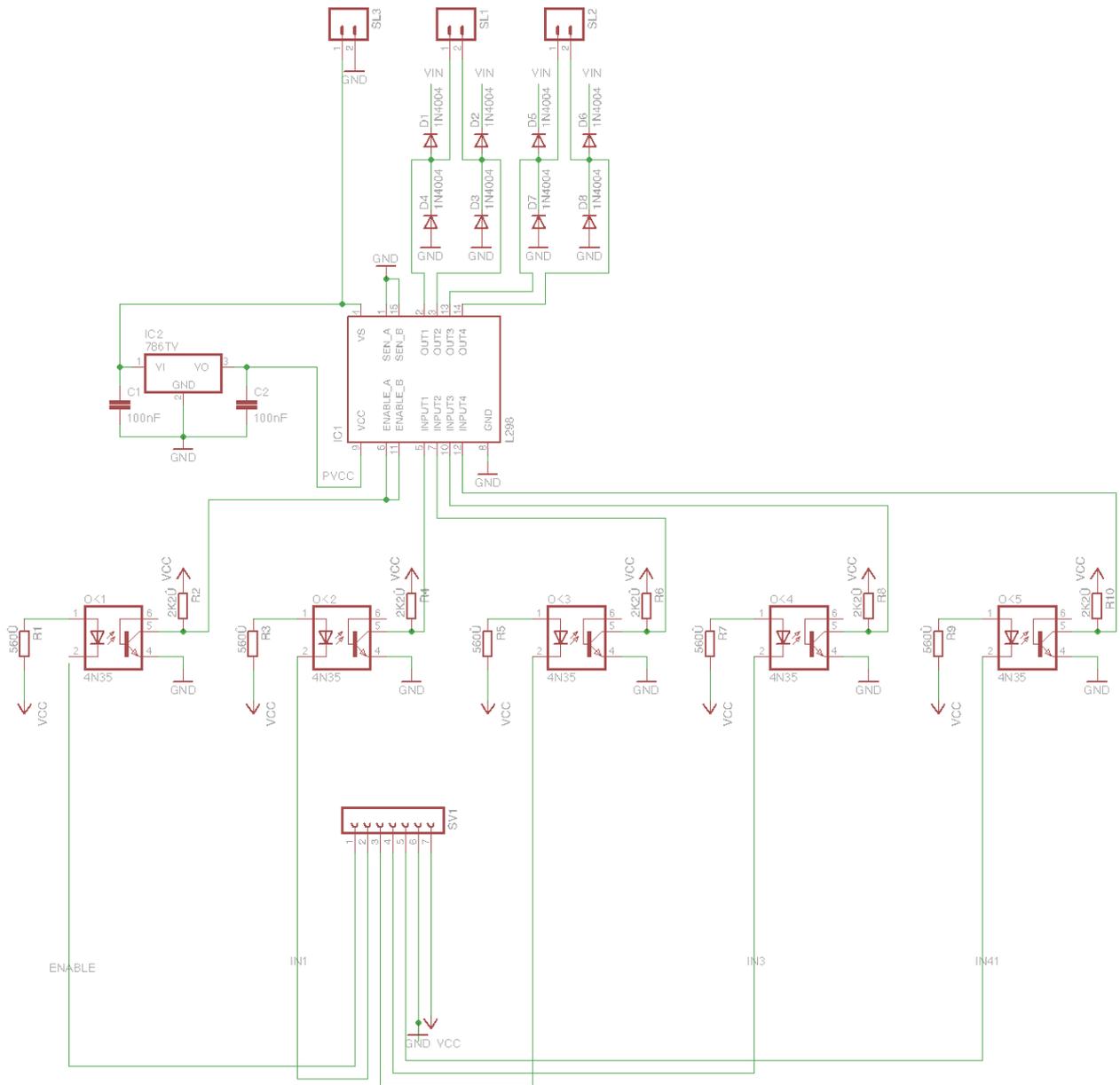
Figura 49: Circuito Impresso do Driver dos Motores.

Fonte: Autoria própria

Figura 50: Esquemático do Driver dos Motores.

7.5 Sensor de efeito Hall

Devido a impossibiliddade de utilização o encoder no carro, utilizamos o sensor de efeito hall. A perda de precisão é relativamente grande. Ao invés de 100 pulsos por volta completa da roda teremos 4. Os imãs acoplados na rodas e o sensor pode ser visto na figura 51. Cada pulso será correspondente a 4 cm de movimento do carro em linha reta.

7.6 Modo automático

No modo automático do carro, foi feito um algoritmo utilizando os dados recolhidos pela câmera de linha. A resposta da câmera é um vetor de tamanho 128 com números analógicos de 0 a 1. O algoritmo consiste dos seguintes passos:

Figura 51: Sensor de efeito hall acoplado ao carro

- Transformar esses numero em 0 ou 1. Onde 1 significa claro e 0 escuro.
- Identificar a faixa preta na imagem (concentração de zeros).
- Contar o número de 1's a esquerda
- Contar o número de 1's a direita
- Tomar decisão.

Para transformar os valores analógicos em digitais é preciso estabelecer um valor limite, que serve de diferenciador entre claro e escuro. Esse valor é determinando segundo o seguinte critério:

$$Valor = V_{MAX} * 0.8$$

Ou seja, é igual a 80% do valor máximo da leitura (mais claro da leitura). As leituras maiores que esse valor são consideradas 1 e as menores 0. Vale salientar que esse valor é calculado novamente a cada leitura.

Após a conversão a faixa preta pode ser identificada na concentração de zeros. O algoritmo de controle considera dez zeros como a faixa preta.

Quanto mais deslocada está a faixa preta do circuito, mais o servo vai virar. A intensidade da curva se baseia no numero de 1's deslocados. Outro fator que deve ser levado em consideração é a velocidade do carro, pois quanto mais veloz o carro mais intensa deve ser a curva.

7.7 Integração

Com todos os componentes funcionando separadamente foi preciso fazer a integração.Esse é um momento muito importante para o projeto. Apesar de parecer simples, grandes complicações podem acontecer. O primeiro passo foi fazer um levantamento dos pinos utilizados por cada componente. E caso componentes estivessem utilizando o mesmo pino, fazer uma realocação. Na Figura 52 podemos ver todos os pinos da placa utilizada em nosso projeto.

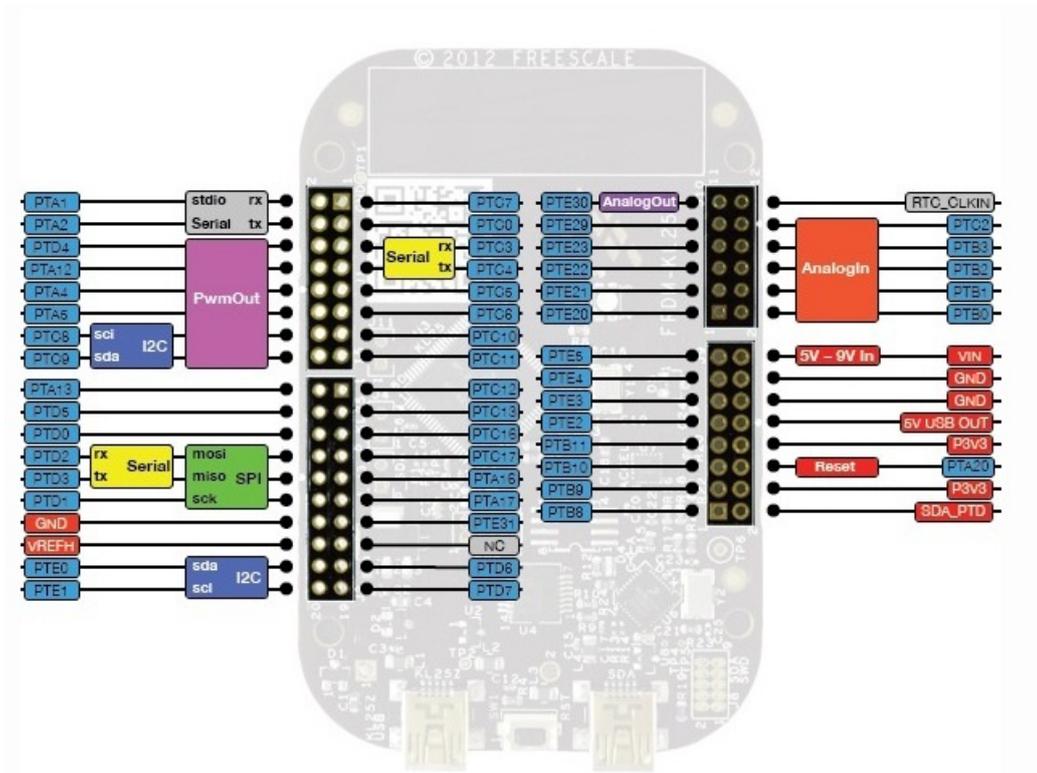



Figura 52: Pinos disponíveis na placa

E na Figura 53 temos a lista de pinos utilizados e o componente.

No geral, não tivemos grandes problemas com a integração. O único contra tempo foi a frequência do PWM utilizada no controle dos motores que precisou ser reduzida, pois o pino de saída não suportava a frequência anterior. Os sensores não tiveram nenhuma alteração no comportamento ao serem ligados todos ao mesmo tempo.

Pino	Componente
PTA5	Motor
PTC8	Motor
PTA12	Motor
PTD4	Motor
PTA13	Motor
PTC9	ServoMotor
PTD3	RF
PTD2	RF
PTC5	RF
PTC17	RF
PTA16	RF
PTA17	RF
PTE1	Acelerômetro
PTE0	Acelerômetro
PTD7	Câmera
PTA4	Câmera
PTD5	Câmera

Figura 53: PinosUtilizados

8 — Cronograma e Custo

Durante o projeto a equipe teve inúmeros problemas no desenvolvimento da tarefas, e isso acabou impactando a execução do cronograma. Após alguns atrasos e trocas de gerente, um novo cronograma foi feito, levando em conta apenas o tempo que restava e as tarefas que ainda poderiam ser completadas. Os cronogramas podem ser vistos no apêndice.

Na tabela 1 vimos o custo previsto em componentes. No decorrer do projeto foi necessário adicionar mais um item a lista, o sensor de efeito hall, devido a mudança do encoder utilizado. Isso fez com que o custo se elevasse levemente. Porém, não ficou longe do programado, como visto na tabela 21. A quantidade de horas trabalhadas e o custo total (tabela 2) não se alteraram.

Tabela 21: Tabela de custos dos componentes atualizada do projeto.

Quantidade	Componente	Custo Estimado
1	Kit de desenvolvimento Freescale (Placa de desenvolvimento MPC5604B) (Chassi do carro) (Servomotor Futaba S-3010) (Ponte H dupla com máx. 5A) (Câmera de linha TAOS 1401)	R$292,00
2	Encoder	R$40,50
1	Acelerômetro e giroscópio (MPU-6050)	R$50,00
1	Bateria VEX 7,2V 2000mA	R$89,00
2	Módulo RF nRF24L01	R$ 20,00
1	Placa de suporte de acrílico	R$30,00
1	Fabricação da PCB de interfaceamento	R$100,00
1	Placa de LEDs	R$60,00
2	Sensor de efeito hall	R$30,00
Custo Total	R$711,50	

9 — Conclusões

9.1 Análise do desenvolvimento

Ao longo do desenvolvimento do projeto a equipe encontrou algumas dificuldades no aspecto gerencial, na escassez de tempo para conclusão das tarefas e no desenvolvimento técnico, a seguir relatadas.

A ausência de uma comunidade de suporte e de solidez, bem como de informação sobre a plataforma de desenvolvimento adotada (Freescale Freedom Board - KL25Z), que é uma novidade no mercado, dificultou a resolução de problemas técnicos encontrados e ocasionou riscos, prejudicando o andamento do projeto.

A maior dificuldade do grupo foi em relação à comunicação. Isto foi verificado quando a equipe não conseguiu realizar uma comunicação bidirecional com o módulo de radio frequência (NRF24L01+) utilizando esta plataforma, mas conseguiu em outras. Este empecilho, que acreditamos ser causado exclusivamente pela placa, foi o que mais consumiu tempo no projeto e precisou ser contornado.

Dentre os riscos ocorridos destacam-se o recebimento de componentes com mau funcionamento, como o driver dos motores e um módulo RF. Também ocorreu a danificação da placa controladora, que não afetou mais fortemente o cronograma, porque a equipe possuía uma placa de reserva.

Outro risco ocorrido no projeto foi a subestimação do tempo e complexidade das tarefas. Isto aconteceu no acoplamento e funcionamento dos encoders, que precisaram ser substituídos por sensores de efeito hall e na implementação do sistema de comunicação.

Houve ainda alguns problemas na gerência. A dificuldade de gerar sinergia entre os membros da equipe e de delegar tarefas fez com que o gerente inicial solicitasse a troca de função com outro membro (com o consentimento do grupo).

Novo problema voltou a ocorrer (algo não planejado nos riscos) com o novo gerente, que não conseguiu apresentar em tempo as tarefas que lhe cabiam, ocasionando a desistência do próprio gerente na metade do semestre, mais especificamente na entrega do terceiro entregável. Outro gerente foi escolhido, sendo este o gerente atual da equipe. Esta troca atrapalhou muito a equipe na parte de organização do cronograma, visto que o gerente anterior não atualizava o mesmo.

Apesar de todos os problemas enfrentados pela equipe, chegou-se a um robô funcional capaz de ser controlado manualmente e capaz de percorrer a pista de forma autônoma utilizando a câmera. A comunicação bidirecional foi estabelecida utilizando um microcontrolador e módulo RF adicionais, e os encoders foram substituídos por um sensor de efeito hall, o que solucionou os dois principais desafios da equipe. O mapeamento também foi realizado, utilizando dados do giroscópio e do sensor de efeito hall.

A equipe não teve nenhum outro problema referente ao planejamento de riscos, sendo a câmera, o acelerômetro, giroscópio e outras tarefas realizadas dentro do prazo.

9.2 Trabalhos futuros

Apesar de a equipe julgar a execução do projeto satisfatória, existem muitos pontos que poderiam ser abordado por equipes futuras na disciplina para a melhoria deste projeto. Podemos destacar entre eles a implementação de uma comunicação bidirecional utilizando somente dois módulos RF, que era o plano inicial do projeto mas que não foi executado devido a problemas técnicos.

Outro campo com grande potencial de melhora é o controle automático do robô. Neste projeto implementamos apenas o controle proporcional, mas outras abordagens poderiam ser utilizadas, como a implementação de um PID (controle proporcional integral e derivativo), o que aumentaria a eficiência no controle do carro.

A forma de detecção da pista também poderia ser melhorada. Na implementação atual aplicamos o algoritmo de controle com base em uma leitura de câmera. É possível utilizar algoritmos de processamento de imagem para juntar leituras consecutivas da câmera de linha e gerar uma imagem bidimensional da pista para efetuar uma detecção da faixa mais precisa.

Também sugerimos a alteração do modo de contagem das rotações das rodas. Utilizamos para isto 4 imãs acoplados em cada roda e utilizamos um sensor de efeito hall para detectarmos os giros. A resolução, entretanto, poderia ser melhorada com outro método.

Referências Bibliográficas

[1] Marshall Brain. Como funcionam os giroscópios.

[2] Atmel Corporation. Atmega328p summary, 2013.

[3] SparkFun Eletronics. St-dualgyro-py-breakout-v11, 2010.

[4] Ligia Figueiredo, Ana Gafaniz, Gustavo Lopes, and Ruben Pereira. Aplicações de acelerômetro.

[5] Roberto Paulo Dias Filho. Protoclo de comunicação i2c.

[6] Invensense. Mpu-6050: Product specification, 2012.

[7] Nordic Semiconductor. nrf24l01: Product specification.

[8] TAOS. 128 x 1 linear sensor array with hold, 2007.

A — Cronograma original

Id		Modo da Tarefa	Nome da tarefa	Duração	Início	Término	Predecessoras	/Jul/13 15 17
1			Início do Projeto	0 hrs	Ter 30/07/13	Ter 30/07/13		
2			**Sistema embarcado**	**111 hrs**	**Ter 30/07/13**	**Qua 18/09/13**	**1**	
3			**Construir placas de suporte de acrílico**	**9 hrs**	**Ter 30/07/13**	**Qui 01/08/13**		
4			Projetar placas	2 hrs	Ter 30/07/13	Ter 30/07/13		
5			Comprar placas de acrílico	4 hrs	Ter 30/07/13	Qua 31/07/13	4	
6			Adaptar as placas de acrílico para o carro	4 hrs	Qua 31/07/13	Qui 01/08/13	5	
7			Montar o carro	4 hrs	Qui 01/08/13	Sex 02/08/13	6	
8			**Acelerômetro/giroscópio (MPU-6050)**	**16 hrs**	**Ter 30/07/13**	**Ter 06/08/13**		
9			Estudar o módulo MPU-6050	4 hrs	Ter 30/07/13	Ter 30/07/13		
10			Estudar o protocolo I2C	4 hrs	Qua 31/07/13	Qua 31/07/13	9	
11			Implementar o protocolo I2C no microcontrolador	4 hrs	Qui 01/08/13	Qui 01/08/13	9;10	
12			Configurar a leitura dos dados do sensor	2 hrs	Sex 02/08/13	Sex 02/08/13	11	
13			Testar o funcionamento	3 hrs	Sex 02/08/13	Ter 06/08/13	12	
14			**Câmera de linha**	**33 hrs**	**Ter 30/07/13**	**Ter 13/08/13**		
15			Acoplar a câmera no carro	2 hrs	Ter 30/07/13	Ter 30/07/13		
16			Estudar o módulo ADC	3 hrs	Ter 30/07/13	Qua 31/07/13	15	
17			Configurar e inicializar o ADC	6 hrs	Qua 31/07/13	Qui 01/08/13	16	
18			Configurar GPIO para gerar CLK e SI	2 hrs	Sex 02/08/13	Sex 02/08/13	17	
19			Implementar algoritmo de detecção de borda	10 hrs	Sex 02/08/13	Qui 08/08/13	17;18	
20			Temporizar as leituras	4 hrs	Qui 08/08/13	Sex 09/08/13	19	
21			Calcular a média das leituras	2 hrs	Sex 09/08/13	Sex 09/08/13	20	
22			Otimizar a altura da câmera para ter uma leitura mais nítida.	4 hrs	Sex 09/08/13	Ter 13/08/13	20;15;21	
23			**Motores DC**	**14 hrs**	**Ter 06/08/13**	**Sex 09/08/13**		
24			Estudar geração de pulsos PWM	8 hrs	Ter 06/08/13	Qui 08/08/13	7;106	
25			Estudar os drivers dos motores	2 hrs	Qui 08/08/13	Qui 08/08/13	24	
26			Gerar pulsos PWM	2 hrs	Qui 08/08/13	Sex 09/08/13	24;25	
27			Testar funcionamento dos motores	2 hrs	Sex 09/08/13	Sex 09/08/13	26	
28			**Placa de LEDs**	**19 hrs**	**Ter 30/07/13**	**Qua 07/08/13**		

Id		Modo da Tarefa	Nome da tarefa	Duração	Início	Término	Predecessor	/Jul/13 15 1
29			Projetar placa de LEDs	4 hrs	Ter 30/07/13	Ter 30/07/13		
30			Comprar o material	2 hrs	Qua 31/07/13	Qua 31/07/13	29	
31			Montar a placa	6 hrs	Qua 31/07/13	Qui 01/08/13	30	
32			Acoplar a placa no carro	2 hrs	Sex 02/08/13	Sex 02/08/13	31	
33			Interfacear a placa com o microcontrolador	6 hrs	Sex 02/08/13	Qua 07/08/13	31;32	
34			**Algoritmo de controle**	**32 hrs**	**Sex 23/08/13**	**Ter 10/09/13**		
35			Estudar algoritmos de controle	20 hrs	Sex 23/08/13	Qua 04/09/13	22;105;111	
36			Projetar algoritmo de controle	10 hrs	Qua 04/09/13	Sex 06/09/13	35;33;21;2	
37			Implementar o algoritmo de controle	2 hrs	Sex 06/09/13	Ter 10/09/13	36	
38			**Servomotor das rodas dianteiras**	**8 hrs**	**Sex 09/08/13**	**Qua 14/08/13**		
39			Gerar o PWM	2 hrs	Sex 09/08/13	Ter 13/08/13	24;27;106	
40			Fazer comandos para controlar os dois PWM	6 hrs	Ter 13/08/13	Qua 14/08/13	39	
41			**Servomotor da câmera**	**12 hrs**	**Qui 15/08/13**	**Qua 21/08/13**		
42			Projetar uma haste e o suporte para o servomotor da câmera	2 hrs	Qui 15/08/13	Qui 15/08/13	40	
43			Pesquisar material para a haste e o suporte	4 hrs	Qui 15/08/13	Sex 16/08/13	42	
44			Montar a haste e o suporte	4 hrs	Sex 16/08/13	Ter 20/08/13	43	
45			Criar funções para controlar o servomotor	2 hrs	Ter 20/08/13	Qua 21/08/13	24;44	
46			**Encoders**	**20 hrs**	**Qui 15/08/13**	**Sex 23/08/13**		
47			Estudar encoder	8 hrs	Qui 15/08/13	Sex 16/08/13	105	
48			Acoplar encoders no ca	4 hrs	Ter 20/08/13	Qua 21/08/13	7;47	
49			Estudar se o processamento da placa freescale é suficiente	2 hrs	Qua 21/08/13	Qua 21/08/13	47;48	
50			Implementar algoritmo para funcionamento do encoder	6 hrs	Qua 21/08/13	Sex 23/08/13	49	
51			**Construir uma PCB para os sensores e os módulos**	**16 hrs**	**Qua 28/08/13**	**Qua 04/09/13**		
57			**Software Embarcado**	**20 hrs**	**Ter 10/09/13**	**Qua 18/09/13**		
58			**Projetar software embarcado**	**12 hrs**	**Ter 10/09/13**	**Sex 13/09/13**	**11;21;26;3**	

Id		Modo da Tarefa	Nome da tarefa	Duração	Início	Término	Predecessoras	/Jul/13 15 17
59			Criar diagrama de casos de uso	2 hrs	Ter 10/09/13	Ter 10/09/13	111	
60			Criar máquina de estados	8 hrs	Qua 11/09/13	Qui 12/09/13	59	
61			Criar diagrama de fluxo de dados	2 hrs	Sex 13/09/13	Sex 13/09/13	60	
62			Implementar o software embarcado	8 hrs	Sex 13/09/13	Qua 18/09/13	59;60;61	
63			**Sistema de comunicação**	**44 hrs**	**Qua 07/08/13**	**Qua 28/08/13**	**1**	
64			**RF**	**25 hrs**	**Qua 07/08/13**	**Sex 16/08/13**		
65	✓		Estudar módulo RF	4 hrs	Qua 07/08/13	Qui 08/08/13	108	
66	✓		Estudar protocolo RF	8 hrs	Qui 08/08/13	Ter 13/08/13	65	
67			Implementar comunicação RF no carro	8 hrs	Ter 13/08/13	Qui 15/08/13	65;66	
68			Implementar comunicação RF na estação base	4 hrs	Qui 15/08/13	Sex 16/08/13	65;66;67	
69			Testar a comunicação entre os RF	1 hr	Sex 16/08/13	Sex 16/08/13	67;68	
70	✓		Definir protocolo de comunicação	8 hrs	Ter 13/08/13	Qui 15/08/13	66	
71			**Implementar o protocolo de comunicação no carro**	**16 hrs**	**Qua 21/08/13**	**Qua 28/08/13**	**70**	
72			Fazer o parser para interpretar as mensagens recebidas	8 hrs	Qua 21/08/13	Qui 22/08/13	70;114	
73			Implementar funções para o envio de dados	8 hrs	Sex 23/08/13	Qua 28/08/13	70;72	
74			**Implementar o protocolo de comunicação na estação base**	**13 hrs**	**Sex 16/08/13**	**Qui 22/08/13**	**70**	
75			Instalar uma biblioteca para a comunicação serial	1 hr	Sex 16/08/13	Sex 16/08/13	87	
76			Fazer um parser para interpretar os dados recebidos	4 hrs	Sex 16/08/13	Ter 20/08/13	70;75	
77			Criar métodos para enviar mensagens	4 hrs	Qua 21/08/13	Qua 21/08/13	70;76	
78			Implementar os métodos de envio de mensagens nos controles da GUI	2 hrs	Qui 22/08/13	Qui 22/08/13	77;86	
79			Mostrar na GUI informações a partir dos dados obtidos	2 hrs	Qui 22/08/13	Qui 22/08/13	76;86;78	
80			**Software da estação-base**	**120 hrs**	**Ter 30/07/13**	**Sex 20/09/13**	**1**	
81			**Projetar o software de alto nível**	**37 hrs**	**Ter 30/07/13**	**Qua 14/08/13**		

Id		Modo da Tarefa	Nome da tarefa	Duração	Início	Término	Predecessor	/Jul/13 15 17
82			Criar o diagrama de casos de uso	4 hrs	Sex 09/08/13	Ter 13/08/13	83	
83			Criar o diagrama de classes	6 hrs	Qua 07/08/13	Sex 09/08/13	33;110	
84			Criar o diagrama de sequência	6 hrs	Ter 13/08/13	Qua 14/08/13	82	
85			Projetar a GUI	2 hrs	Ter 30/07/13	Ter 30/07/13		
86			Implementar a GUI	2 hrs	Ter 30/07/13	Qua 31/07/13	85	
87			Implementar o controle básico do software	4 hrs	Qui 15/08/13	Sex 16/08/13	82;83;84;1	
88			**Plotar o gráfico do percurso em 2D**	**8 hrs**	**Sex 23/08/13**	**Qua 28/08/13**	**87**	
89			Criar métodos dummies para gerar informações da odometria	1 hr	Sex 23/08/13	Sex 23/08/13	9;47;79	
90			Criar rotinas para realizar a plotagem do gráfico a partir do modelo	6 hrs	Sex 23/08/13	Ter 27/08/13	89	
91			Testar a plotagem do gráfico	1 hr	Qua 28/08/13	Qua 28/08/13	90	
92			**Plotar o gráfico do percurso em 3D**	**9 hrs**	**Qua 28/08/13**	**Sex 30/08/13**	**88**	
93			Instalar uma biblioteca para fazer desenho 3D	2 hrs	Qua 28/08/13	Qua 28/08/13	91	
94			Criar rotinas para realizar a plotagem do gráfico a partir do modelo	6 hrs	Qua 28/08/13	Sex 30/08/13	93	
95			Testar a plotagem do gráfico	1 hr	Sex 30/08/13	Sex 30/08/13	94	
96			Implementar controles do carro	10 hrs	Sex 30/08/13	Qui 05/09/13	71;74;95	
97			**Odometria**	**116 hrs**	**Qua 31/07/13**	**Sex 20/09/13**		
98			Pesquisa sobre odometria	20 hrs	Qua 31/07/13	Qui 08/08/13	86	
99			Pesquisar métodos para integrar dados dos sensores	40 hrs	Ter 20/08/13	Qui 05/09/13	70;9;47	
100			Implementar a odometria	10 hrs	Qui 05/09/13	Qua 11/09/13	99;98	
101			Testar o funcionamento do sistema de odometria	2 hrs	Sex 20/09/13	Sex 20/09/13	90;2;100;1	
102			Ajustar a odometria	20 hrs	Qua 11/09/13	Qui 19/09/13	100	
103			**Documentação**	**113 hrs**	**Sex 02/08/13**	**Qua 25/09/13**		
104			**Kit Freescale**	**24 hrs**	**Sex 02/08/13**	**Qua 14/08/13**		
105			Câmera de linha	4 hrs	Qua 14/08/13	Qua 14/08/13	14	
106			Chassi + Bateria	2 hrs	Sex 02/08/13	Sex 02/08/13	7	
107			**Sensores**	**43 hrs**	**Ter 06/08/13**	**Sex 23/08/13**		

Id		Modo da Tarefa	Nome da tarefa	Duração	Início	Término	Predecessor	/Jul/13 15	17
108			MPU-6050	4 hrs	Ter 06/08/13	Qua 07/08/13	8		
109			Encoders	2 hrs	Sex 23/08/13	Sex 23/08/13	46		
110			Placa de LEDs	2 hrs	Qua 07/08/13	Qua 07/08/13	28		
111			Servomotores	2 hrs	Qua 21/08/13	Qua 21/08/13	38;41		
112			**Comunicação**	**3 hrs**	**Ter 20/08/13**	**Ter 20/08/13**			
113			RF	2 hrs	Ter 20/08/13	Ter 20/08/13	65;66;69		
114			Protocolo	1 hr	Ter 20/08/13	Ter 20/08/13	70;113		
115			**Software de alto-nível**	**85 hrs**	**Qui 15/08/13**	**Ter 24/09/13**			
116			Diagramas	2 hrs	Qui 15/08/13	Qui 15/08/13	82;83;84		
117			Manual de usuário	6 hrs	Qui 05/09/13	Sex 06/09/13	87;85;95;9		
118			Algoritmo de Odometria	2 hrs	Ter 24/09/13	Ter 24/09/13	97		
119			**Software Embarcado**	**7 hrs**	**Qua 18/09/13**	**Sex 20/09/13**	**62**		
120			Diagramas	2 hrs	Qua 18/09/13	Qui 19/09/13			
121			Algoritmo de controle	2 hrs	Qui 19/09/13	Qui 19/09/13	120		
122			Sensores	3 hrs	Qui 19/09/13	Sex 20/09/13	121		
123			PCB	8 hrs	Qua 04/09/13	Sex 06/09/13	51		
124			Integração	4 hrs	Ter 24/09/13	Qua 25/09/13	104;107;11		
125			**Entregáveis**	**127 hrs**	**Ter 30/07/13**	**Qua 25/09/13**			
126			Primeiro	0 hrs	Qua 07/08/13	Qua 07/08/13			
127			Segundo	0 hrs	Qua 21/08/13	Qua 21/08/13			
128			Terceiro	0 hrs	Qua 04/09/13	Qua 04/09/13			
129			Quarto	0 hrs	Qua 18/09/13	Qua 18/09/13			
130			Entrega do Projeto	0 hrs	Qua 25/09/13	Qua 25/09/13			

B — Novo cronograma

Id	Nome da tarefa	Duração	Início	Término	/Jul/13 15	17	19	21/J 21
1	**Sistema embarcado**	**94 hrs**	**Dom 08/09/1**	**Sex 11/10/13**				
2	Montar o carro	4 hrs	Dom 08/09/1	Dom 08/09/1				
3	**Câmera de linha**	**27 hrs**	**Qua 11/09/13**	**Qua 18/09/13**				
4	Acoplar a câmera no carro	20 hrs	Qua 11/09/13	Sáb 14/09/13				
5	Implementar algoritmo de detecção de borda	10 hrs	Sáb 14/09/13	Qua 18/09/13				
6	Temporizar as leituras	4 hrs	Sáb 14/09/13	Sáb 14/09/13				
7	Calcular a média das leituras	2 hrs	Sáb 14/09/13	Sáb 14/09/13				
8	**Motores DC**	**6 hrs**	**Qua 11/09/13**	**Qui 12/09/13**				
9	Projeto placa de circuito impresso	2 hrs	Qua 11/09/13	Qua 11/09/13				
10	Fazer placa de circuito impresso	3 hrs	Sáb 14/09/13	Sáb 14/09/13				
11	Testar funcionamento da placa de circuito impresso	1 hr	Ter 17/09/13	Ter 17/09/13				
12	**Algoritmo de controle**	**19 hrs**	**Sáb 14/09/13**	**Sex 20/09/13**				
13	Projetar algoritmo de controle	10 hrs	Sáb 14/09/13	Qui 19/09/13				
14	Implementar o algoritmo de controle	4 hrs	Qui 19/09/13	Sex 20/09/13				
15	**Encoders**	**11 hrs**	**Ter 10/09/13**	**Qui 12/09/13**				
16	Acoplar encoders no carro	4 hrs	Ter 10/09/13	Qua 11/09/13				
17	Estudar se o processamento da placa freescale é suficiente	2 hrs	Qua 11/09/13	Qua 11/09/13				
18	Caracteristicas Encoder.	4 hrs	Ter 10/09/13	Qua 11/09/13				
19	Implementar algoritmo para funcionamento do encoder	2 hrs	Qua 11/09/13	Qui 12/09/13				
20	**Construir uma PCB para os sensores e os módulos**	**12 hrs**	**Qua 11/09/13**	**Sáb 14/09/13**				
21	Fazer esquemático	4 hrs	Qua 11/09/13	Qui 12/09/13				
22	Imprimir molde e passar para placa	4 hrs	Qui 12/09/13	Sex 13/09/13				
23	Acoplar PCB no carro	2 hrs	Sex 13/09/13	Sáb 14/09/13				
24	Testar os sensores com a placa impressa	2 hrs	Sáb 14/09/13	Sáb 14/09/13				
25	**Software Embarcado**	**20 hrs**	**Sex 20/09/13**	**Qui 26/09/13**				
26	**Projetar software embarcado**	**12 hrs**	**Sex 20/09/13**	**Ter 24/09/13**				
27	Criar diagrama de casos de uso	2 hrs	Sex 20/09/13	Sex 20/09/13				
28	Criar máquina de estados	8 hrs	Sex 20/09/13	Seg 23/09/13				
29	Criar diagrama de fluxo de dados	2 hrs	Seg 23/09/13	Ter 24/09/13				
30	Implementar o software embarcado	8 hrs	Ter 24/09/13	Qui 26/09/13				
31	**Sistema de comunicação**	**34 hrs**	**Dom 08/09/1**	**Qua 18/09/13**				
32	**RF**	**27 hrs**	**Dom 08/09/1**	**Sáb 14/09/13**				
33	Implementar comunicação RF no carro	27 hrs	Dom 08/09/13	Sáb 14/09/13				

Id	Nome da tarefa	Duração	Início	Término	/Jul/13			21/J
					15	17	19	21
34	Testar a comunicação entre os RF	1 hr	Sáb 14/09/13	Sáb 14/09/13				
35	**Implementar o protocolo de comunicação no carro**	**11 hrs**	**Sáb 14/09/13**	**Qua 18/09/13**				
36	Fazer o parser para interpretar as mensagens recebidas	8 hrs	Sáb 14/09/13	Qua 18/09/13				
37	Implementar funções para o envio de dados	8 hrs	Qua 18/09/13	Sex 20/09/13				
38	**Software da estação-base**	**37 hrs**	**Sáb 07/09/13**	**Ter 17/09/13**				
39	**Odometria**	**50 hrs**	**Sáb 07/09/13**	**Sex 20/09/13**				
40	Pesquisa sobre odometria	23 hrs	Sáb 07/09/13	Qui 12/09/13				
41	Implementar a odometria	10 hrs	Qui 12/09/13	Sáb 14/09/13				
42	Testar o funcionamento do sistema de odometria	4 hrs	Sex 20/09/13	Sex 20/09/13				
43	Ajustar a odometria	20 hrs	Sáb 14/09/13	Sex 20/09/13				
44	**Entregáveis**	**46 hrs**	**Qui 12/09/13**	**Qua 25/09/13**				
45	Quarto	15 hrs	Sáb 14/09/13	Qua 18/09/13				
46	Entrega do Projeto	15 hrs	Seg 23/09/13	Qua 25/09/13				

C — Diagrama de Sequência

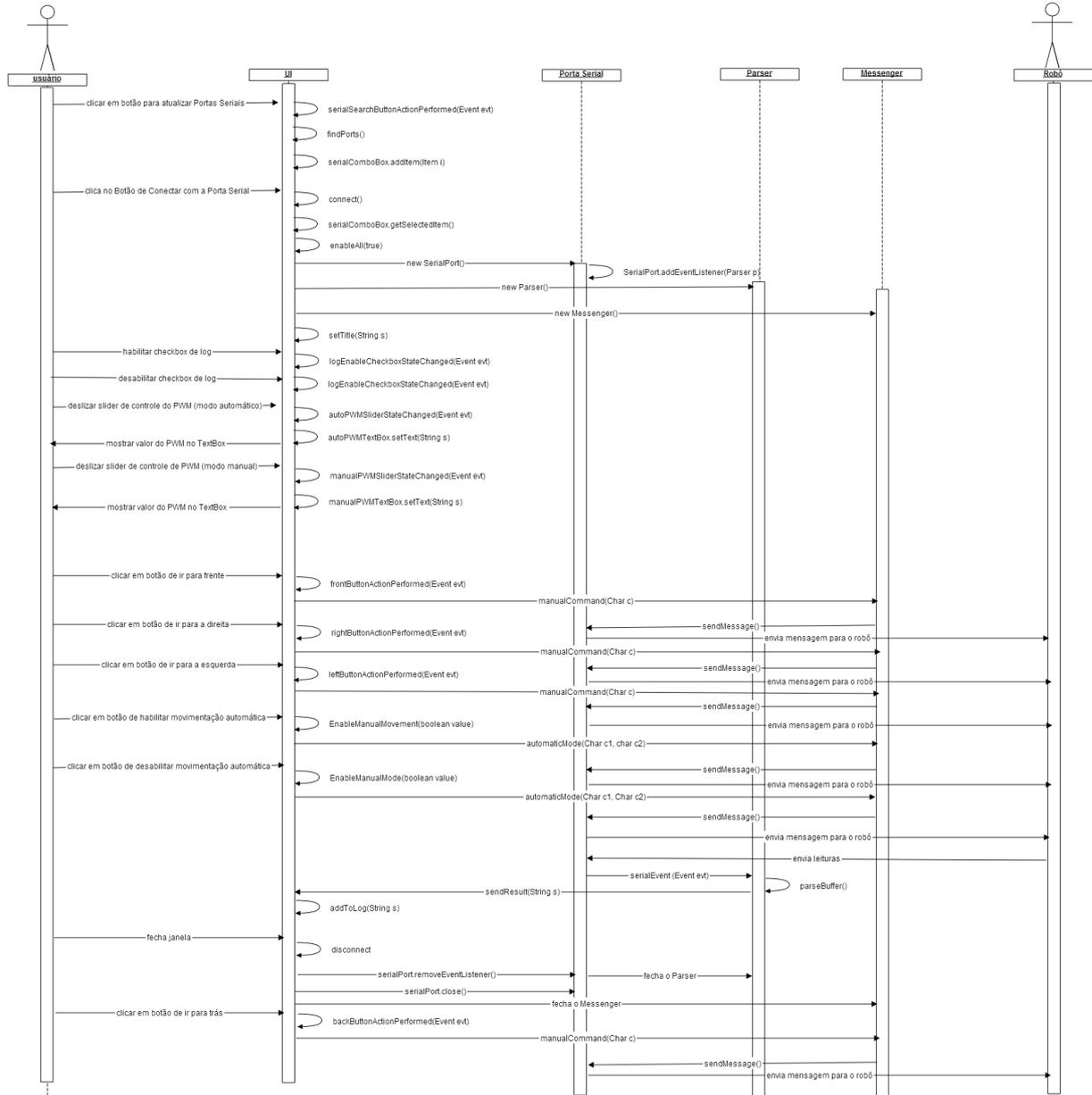



Figura 54: Diagrama de Sequência.